appGATE™
Network Security

AppGate Security Server, Version 8.0.4

Security Target

Document Version: 2.9

Date: 2008-04-10

Contents

Index of Illustrations and Tables

Index of Tables

Index of Figures

1 Introduction

1.1 ST Identification

ST Title:	AppGate Security Server, version 8.0.4, Security Target
Product Name:	AppGate Security Server
Product Version:	Version 8.0.4
Assurance level:	EAL2+
CC Version:	2.3, as of August 2005
ST author:	Sebastian Mayer
ST publication date:	2008-04-10
ST Version:	2.9

Keywords: application gateway, VPN, access control (Role- and rule-based system), network security, activity logging, remote access, authentication methods

1.2 ST Overview

This document is the Security Target (ST) for the AppGate Security Server, version 8.0.4 running on the Sun Solaris 10 operating system and delivered as an appliance in a turnkey package. The AppGate Security Server is an application level gateway. It controls user access to protected resources through a flexible rule system. All user traffic to the TOE is encrypted using SSHv2. Central remote administration of the TOE is performed through an easy-to-use graphical user interface (GUI).

The ST contains a description of the security objectives and the requirements, as well as the necessary functional and assurance measures provided by the TOE. The ST provides the basis for the evaluation of the TOE according to the Common Criteria for Information Technology Security Evaluations (CC).

1.3 CC Conformance Claim

This ST is CC Part 2 extended and CC Part 3 conformant, with the assurance level of EAL2 augmented with ALC_FLR.1. The ST was extended with FCS_RNG.1 The Security Target follows the structure given in Part 1 of the Common Criteria, using the guidance from ISO/IEC JTC 1/SC 27 N 2449 “Information technology – Security techniques – Guide for the production of protection profiles and security targets” ([GPPS]).

This ST does not claim conformance to any existing Protection Profile (PP).

1.4 Strength of Function Claim

The TOE contains one function that is dependent on probability: authentication via password. The minimum strength of function claimed for this function is SOF-medium. No strength of function analysis is performed for the cryptographic algorithms supported by the TOE as well as the process of the generation of the keys used by those cryptographic algorithms.

1.5 ST Content and Organization

The ST has been structured in accordance with [CC] Part 1 and [CEM]. The main sections of the ST are the TOE description, TOE security environment, security objectives, IT security requirements, rationale and annexes.

The TOE description provides general information about the TOE, serves as an aid to understanding the nature of the TOE and its security functionality, and provides context for the ST's evaluation.

The TOE security environment describes security aspects of the environment in which the TOE is to be used and the manner in which the TOE is to be employed. The TOE security environment includes:

a. assumptions regarding the TOE's intended usage and environment of use

b. threats relevant to secure TOE operation

c. organizational security policies with which the TOE must comply

The security objectives reflect the stated intent of the TOE. They pertain to how the TOE will counter identified threats and how it will cover identified organizational security policies and assumptions.

Each security objective is categorized as being for the TOE or for the environment.

The security requirements section provides detailed requirements, in separate subsections, for the TOE and its environment.

The IT security requirements are subdivided as follows:

a. TOE Security Functional Requirements

b. TOE Security Assurance Requirements

c. Security requirements for the IT environment

The TOE summary specification addresses the security functions that are represented by the TOE to answer the security requirements.

The rationale presents evidence that the ST is a complete and cohesive set of requirements and that the TOE provides an effective set of IT security countermeasures within the security environment. The rationale is in two main parts. First, a security objectives rationale demonstrates that the stated security objectives are traceable to all of the aspects identified for the TOE and the TOE security environment and are suitable to cover them. Then, a security requirements rationale demonstrates that the security requirements for the TOE and the TOE environment are traceable to the security objectives and are suitable to meet them.

The appendix contains a list of abbreviations as well as a glossary for this ST.

1.6 Related Standards and Documents

[CC] Common Criteria (CC) for Information Technology Security Evaluation, August 2005, Version 2.3:

- Part 1: Introduction and general model. August 2005. Version 2.3. CCMB-2005-08-001
- Part 2: Security functional requirements. August 2005. Version 2.3. CCMB-2005-08-002
- Part 3: Security Assurance Requirements. August 2005. Version 2.3. CCMB-2005-08-003

[CEM] Common Methodology for Information Technology Security Evaluation, August 2005, Version 2.3. CCMB-2005-08-004.

[JIL] Joint Interpretation Library: Collection of Developer Evidence. Version 1.0, as of: August 2000.

[GPPS] ISO/IEC TR 15446:2004(E), Guide for the production of Protection Profiles and Security Targets, First edition 2004-07-01.

[MAN] AppGate Security Server manual version 8.0.4

[SECSERV] AppGate Security Server version 8.0

[RFC 4251] SSH Protocol Architecture January 2006

[RFC 4253] SSH Transport Layer Protocol January 2006

[RFC 4252] SSH Authentication Protocol January 2006

[RFC 4254] SSH Connection Protocol

[RFC 4256] Generic Message Exchange Authentication for the Secure Shell Protocol SSH), January 2006

[RFC 2104] HMAC: Keyed-Hashing for Message Authentication February 1997

[FIPS 180-2] US Secure Hash Algorithm 1 (SHA1) August 2002

[FIPS 197] Advanced Encryption Standard (AES) November 2001

[FIPS 186-2] Digital Signature Standard (DSS) January 2000

[SPP] Security IC Platform Protection Profile, Version 1.0 as of 15.06.2007

2 TOE Description

This section describes the Target of Evaluation (TOE) in terms of the class of product, the operational environment, and the provided security functionality.

Chapter 2.1 provides an introduction to the AppGate Security Server.

Chapter 2.2 describes the architecture the AppGate Security Server.

Chapter 2.3 introduces the TOE and its definition scope.

Chapter 2.4 gives an overview of the security functionality of the TOE.

Chapter 2.5 describes the evaluated configuration of the TOE.

Chapter 2.6 describes the security roles of the TOE.

2.1 Introduction

The AppGate Security Server is a centrally-managed application level VPN gateway that separates networks of different security levels by granting or denying access from clients residing in networks of a lower security level (called “unprotected network” in the following description) over secure communication channels to resources / applications in networks of a higher security level (called “protected network” in the following description) on the basis of granular access rules. For example, clients residing in a LAN within a corporate network , separated from internal networks consisting of databases with highly secure information only accessible by users with special rights.

In all possible architectures, it is essential that all user traffic must pass the AppGate Security Server and is subject to access control before it reaches protected resources, mostly residing in a separate network. Figure 1 shows the general network structure:

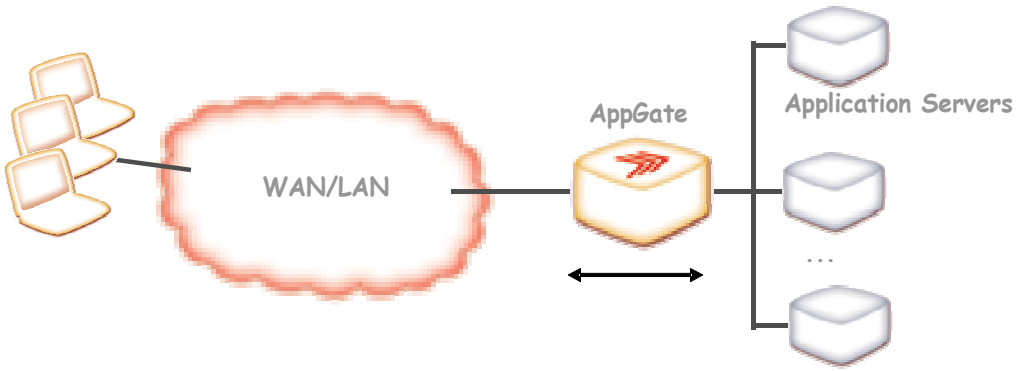

Figure 1: AppGate Security Server mediating access of users in the WAN/LAN unprotected network (to the left) to network resources in the protected network (to the right)

The communication ways between the clients and the AppGate Security Server across the unprotected network are secured through server authentication, secure key exchange, data encryption and the ability to detect loss of data integrity, using SSHv2 ([RFC 4251]).

Multiple user authentication methods are supported by the AppGate Security Server, e.g., using passwords, certificates, SecurID-tokens, or requesting external authentication servers such as LDAP, Radius, or SecurID servers (RSA ACE/Server). For this evaluation, only password and SecurID authentication are allowed. These authentication mechanisms are triggered by the SSH protocol.

After the successful authentication of the user, the AppGate Security Server checks the authorization of the user to access network resources in the protected network. A flexible

rule system controls the user's access to the protected network resources, using a configurable structure of roles, services, and service groups. Access decisions can be based on many attributes of users, server, and environmental information.

After successful authentication and authorization, the AppGate Security Server enables the user to access resources in the protected network. The communication between the Appgate Security Server and the resources in the protected network, has the form of the chosen service by the user, and is not necessarily encrypted. The Appgate Security Server always works as a proxy towards the resources in the protected network. All communications between the TOE and the resources in the protected network are not subject to the evaluation.

All remote user tasks (either using protected resources or administering the AppGate Security Server) can be performed through an easy-to-use GUI.

Further, all kinds of remote user activity can be logged, and alarms on defined events can be generated.

The AppGate Security Server contains a wide range of functions. Its main functionalities can be summarized as follows:

- secure communication ways
  - server authentication
  - confidentiality through data encryption of user network traffic
  - integrity verification of user network traffic
- user authentication supporting different authentication methods
- authorization
  - rights management for access to resources
- central administration
- easy access and administration through a GUI
- logging of activities and generating alarms

The following functionality is also part of the AppGate Security Server, but were not evaluated:

- firewall functionality
- ability to work in a server cluster for high availability and scalability
- secure instant-messaging
- secure printing
- integration of all kinds of network services like Web/FTP, NetBIOS shares, single client/server commands

2.2 AppGate Architecture

The overall AppGate Architecture is implemented as a client server architecture with the AppGate Security Server acting as an application level gateway, separating unprotected networks on which the clients reside, from protected networks.

2.2.1 Server

The AppGate Security Server is a software product delivered as an appliance with Sun Solaris OS and a Sun hardware platform shipped from the factory as a single pre-installed unit. The server is implemented as a set of application layer programs and daemons. These agents communicate internally via TCP, UDP, pipes, the file system, and secmsg, a

proprietary AppGate messaging system. The AppGate Security Server typically separates networks of different security levels as described above. All traffic must pass the Security Server.

The Server requires the use of SSHv2 for connections between remote users and the Server on the unprotected network.

The Server identifies and authenticates remote users. For this task, the Server can use a local database (flat file), or external databases like LDAP or SecurID server (the evaluated configuration requires that user accounts must reside in the local database).

Based on the user data, authentication method, and environmental information, the Server controls access to the protected network, or to itself in case of a remote administrator connecting to the AppGate Security Server.

All kinds of client access is mediated by the server. The server is running on a dedicated machine with no other applications running under the control of a user. This ensures that normal users have no direct access to server resources, e.g, configuration and audit files.

2.2.2 Client

The AppGate client software (not part of the evaluation) is deployed on the user machines and provides the ability to connect to the AppGate Security Server and to establish a secure channel through an unprotected network (VPN). For client applications using services behind the AppGate server there is usually no need to make any configuration changes to them on the client machines , e.g., a RDP (Remote Desktop Protocol) client application uses the established VPN channel without having to change the configuration of the RDP client application.

There are two general types of clients: the AppGate Client for normal users, and the AppGate Console for administrators of the AppGate Security Server. Both of these clients can be either installed or used via Java Web Start technology.

Additional clients for mobile devices or clients using signed Java applets exist. It is also possible to connect to the AppGate Security Server using a normal OpenSSH client. However, these three client types have reduced functionality compared to the AppGate Client/Console.

For the evaluated configuration, only clients installed from the AppGate installation media should be used.

2.3 TOE Definition Scope

The Target of Evaluation (TOE) is a software-only product and is limited to the server part of the AppGate Architecture. This is illustrated in the Figure 2 showing the scope of the TOE and other parts that may be used by the TOE in its evaluated configuration. The TOE rely on the operating system and hardware for its operation, but may also rely on external servers for user authentication, e.g. an RSA ACE/Server that provides SecurID authentication.

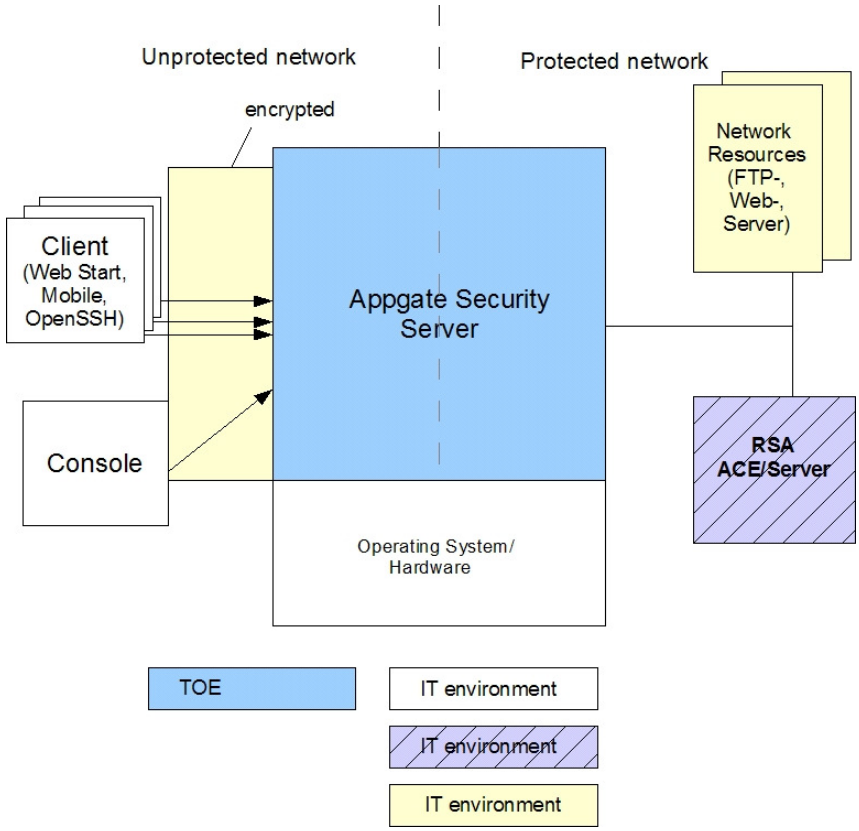

Figure 2: TOE definition scope

The physical boundary of the TOE is limited to the AppGate Security Server 8.0.4 installed as an appliance on Sun hardware and running the Sun Solaris 10. Along with the appliance is a CD or USB stick containing the PDF version of the user and administrator manual "AppGate Security Server, Version 8.0.4" that also contains instructions for bringing the TOE into the evaluated configuration.

Not part of the TOE are:

- underlying hardware and operating system
- clients or any other client components such as the Personal Firewall or the IP Tunneling Driver
- protected resources on the network behind the TOE
- external third-party authentication services/servers as part of the protected resources

Logically the TOE consists of:

- VPN functionality through SSHv2
- password authentication
- SecurID authentication management

- access control
- logging

The following AppGate Security Server components and functionality are excluded:

- IP filter (extension of the operating system) and the management functions of the built-in firewall
- support for authentication methods other than password and SecurID authentication
- support for secure printing, roaming and secure messaging
- generation and destruction of host keys (public key of the AppGate Security Server)
- server authentication using X.509 certificates (instead, public server keys are pre-installed on the client)
- clustering functionality

While SecurID authentication management is part of the TOE, the process of the SecurID authentication itself, using an external SecurID server, is part of the IT environment. Evaluation of the SecurID authentication management verifies that the SecurID authentication data of the user is correctly and securely transferred from the user to an external SecurID server, and that the authentication decision of this SecurID server is correctly integrated into the authorization process of the TOE.

2.4 Overview of TOE Security Functions

This section outlines the security functionality of the TOE. The security functions include:

- secure communication between users and the TOE through server authentication, data encryption and data integrity verification using SSHv2
- user identification and authentication
- Role- and rule-based access control to services
- logging and audit functions

These functions can be divided in those that are provided by the TOE implementing SSHv2, and those that are provided by the TOE independent from SSHv2.

2.4.1 Security functions provided by SSHv2

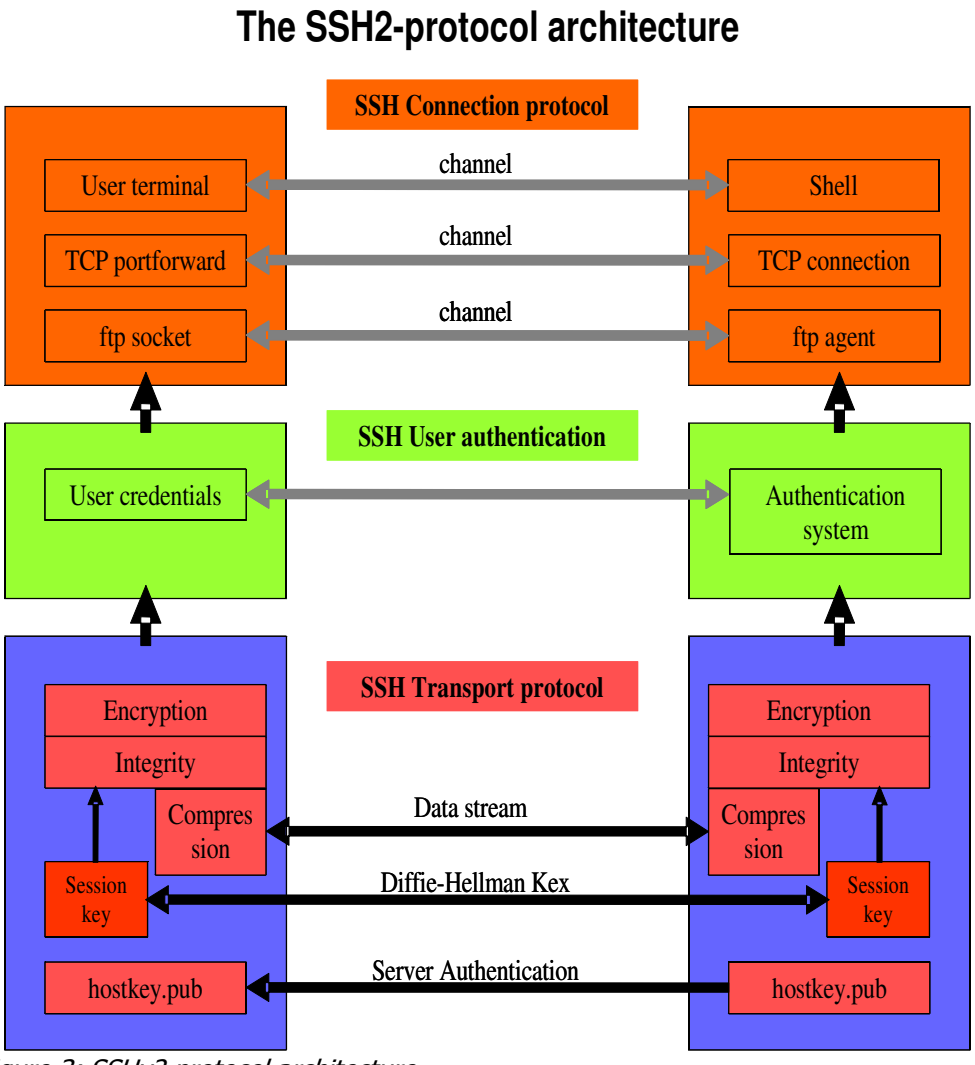

Figure 3: SSHv2 protocol architecture

The three layers of the SSHv2 architecture [RFC 4251] that correspond to different TOE functions are shown in figure 3.

The SSH Transport Protocol [RFC 4253] forms the basis and provides secure communications by means of server authentication, confidentiality and integrity between remote users and the TOE.

The SSH User Authentication Protocol [RFC 4252] and the Generic Message Exchange Authentication [RFC 4256] provide the client authentication for password authentication and SecurID authentication (one-time passwords), respectively. Relying on the underlying SSH Transport Protocol, the identification and authentication data is securely transmitted.

The SSH Connection Protocol [RFC 4254] has been designed to run on top of the SSH User Authentication Protocol. It provides, e.g., interactive login sessions, remote execution of commands, and forwarded TCP/IP connections. All of these channels are multiplexed into a single encrypted tunnel. The SSH Connection Protocol provides no new security functionality in the context of the TOE, because the SSH Communication Protocol does not provide any of the security functions described in this section 2.4.

2.4.1.1 Secure communication

The SSH Transport Protocol in the bottom of Figure 3 provides secure communication channels between user clients and the TOE.

The TOE performs signatures using a private server key to authenticate to the users, preventing man-in-the-middle attacks (public server key must be pre-installed on the client side).

Symmetric keys for encryption/decryption and message authentication are provided using the Diffie-Hellman key exchange. The quality of the encryption keys and the message authentication keys used within Diffie-Hellman (DH parameters) are ensured by using random numbers of good quality when computing such keys. This random number generator is part of the TOE environment.

After the successful key exchange, a message authentication code using the respective authentication key and a cryptographic hash function, is calculated over the payload of the transmitted packets , providing the ability to detect loss of integrity and authenticity on the receiving side. The verification of hashes or authentication codes is done through also calculating the hash or authentication code and then comparing it to the received value. Confidentiality is ensured by encrypting the payload with the encryption key that is securely created during the Diffie-Hellman key exchange. These actions apply to both client and server.

2.4.1.2 User identification and authentication

User identification and authentication are built on top of the SSH Transport Protocol (see figure 3). Using SSHv2 the user credentials are transmitted securely to the TOE. While a large number of different authentication methods are available, only two authentication methods are supported by the evaluated configuration, password authentication and SecurID.

In both cases the TOE first looks up the user name in the local database. In case of password authentication, the authentication is performed according to [RFC 4252], and the TOE compares the provided user identity and password with the user data in the local database. While in the case of SecurID authentication, the authentication is performed according to [RFC 4256], and the TOE forwards the provided user id, the personal identification number, and a random number generated by the SecurID token to an SecurID server and requests it to decide on the user's authenticity. The user data resides in this case both on the TOE and the SecurID server (data necessary for SecurID authentication).

2.4.2 Security functions independent from SSHv2

2.4.2.1 Rights management and authorization

Access to resources is controlled by Role- and rule-based rights management. Users can have one or more roles assigned to them. For each role, a group of services, possibly organized within folders, can be defined. Services are a collection of components that are logically grouped together to perform one task. A component is the smallest entity in the hierarchy and enables a certain activity, like Client commands, Proxy setup, IP access to resources in the protected network, or administrative access on the TOE. Each component has configurable parameters that are typical for the activity that the component represents. Different instances of a component with different parameters can be created to meet the needs of users. Figure 4 shows the associations of the objects mentioned:

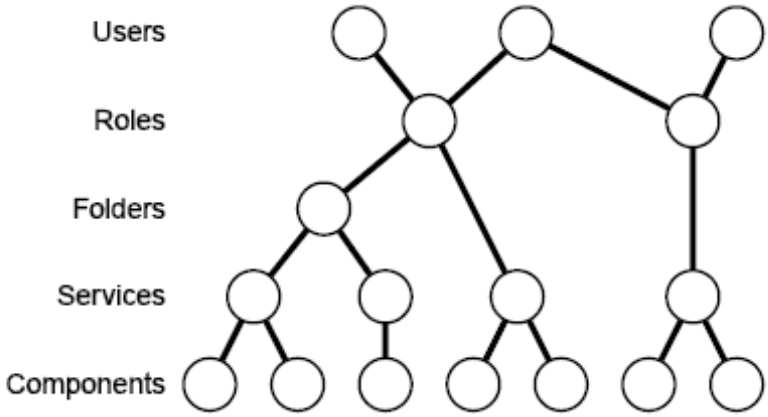

Figure 4: Hierarchy of objects that control and represent access to protected network resources

As shown in the figure above, the objects roles, folders, services and components build an object-tree. Access to the objects in this tree is controlled via the following Security Function Policy (SFP):

Role- and rule-based access SFP
A user who has access to a role also has access to all folders and services that are associated to that role. The administrator configures roles to which a user has access. A user can use one role per session.

After a user is granted access to a specific role, access rules are used to make further access decisions based on additional information, like the user's authentication method, the user's IP address, or other environmental information. Access rules allow verification of access on a more granular basis, because they can be attached to roles or to associations of roles/folders, roles/services, or folders/services. Figure 5 clarifies this point:

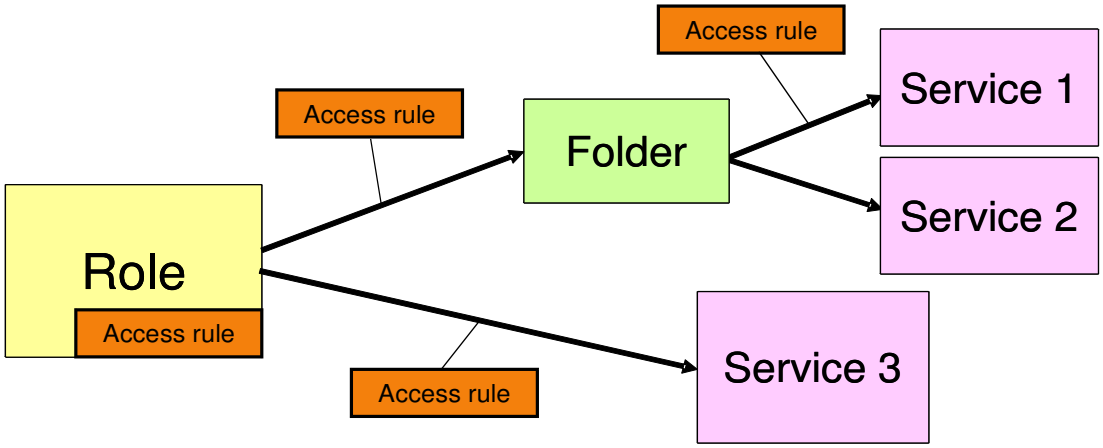

Figure 5: Use of access rules

For example, whether a user has access to “Service 1” in figure 5, depends on a “positive” access decision of all three access rules in the path from the role to this service. The Role- and rule-based access SFP follows the "least privilege" principle.

Role with associated objects can also be seen as static access control model, where an administrator defines which role a user is allowed to access, together with all objects defined under the the role.

In contrast, access rules can be seen as dynamic access control model, where access to an object is made just in time of the request, where user session attributes like the IP address are evaluated.

Administrative access
Administrative access to the TOE is based on the SFP described above. However, only if a user has access to the "Administration access" component, he is allowed to open the administrative view within the AppGate Console and manage the TOE.

To sum up the access control policy of the AppGate Security Server, here is another example that involves the two access SFPs and the "Administration access" component configuration:

Whether a user has administrative access to the TOE depends first on the roles to which he has access. After the user selects one of the allowed roles, all folders and services that were associated to the role are displayed and can be used, presumed that no access rules explicitly deny access to specific folders and services. If one of these services consist of the "Administration access" component, the user is allowed to open the administrative view that presents all manageable objects, with the help of the AppGate Console.

2.4.2.2 Auditing

The following security-relevant events are logged:

- login and logout
- administrative actions
- commands started or finished

Log file contents can be displayed and actions can be defined in case the audit trail exceeds a defined limit.

2.4.2.3 Security management

Administrators use the AppGate Console to manage all security functions through a graphical user interface (GUI), including:

- audit trail limits and alarms
- authentication methods
- user accounts
- access rules
- roles, folders, services and components

All users that have access to the "Administration access" component are administrators of the TOE.

2.5 Evaluated Configuration

The following configuration of the TOE is considered as evaluated configuration:

- user accounts must reside in the local database (internal user database)
- the allowed user authentication methods are:
  - Password
  - SecurID (which implies that a SecurID server is configured to perform the SecurID authentication on behalf of the TOE)
- Roaming, secure printing, and secure messaging is not allowed
- Clustering functionality is not allowed

2.6 Users of the TOE and Security Roles

Authenticated users of the TOE are either normal remote users or administrative remote users. Both are referred to as remote users. The TOE distinguishes between normal users and administrative users (administrators). These two security roles are different from what are called “roles” in the administrator's GUI. The AppGate roles are used to group services and components together – there is no pre-defined administrative role. However, an administrative user is defined by assigning the "Administration access" component to the role to which this user has access. Any number of administrative users can be defined in this way.

The administrative component recognizes two different access models: unrestricted access and user account administration only. However, in this evaluation, no difference is made between these two modes, assuming that the administrator always has unrestricted access.

An administrator with unrestricted administrative access has full administrative rights to the TOE. Such an administrator also has unrestricted access to the underlying operating system, because the administrator can configure the “File transfer” component, that enables him to modify all files in the operating system with root privileges via the AppGate Console. Access to the configuration files, log files, and database files is only allowed indirectly by using the TOE and the AppGate Console.

Throughout this document, the term “roles” is always used to describe the role objects that are visible in the administrator's GUI.

3 TOE Security Environment

The assumptions made and the threats addressed are summarized in the following section.

3.1 Assumptions

The following conditions are assumed to exist in the TOE operational environment. These assumptions include essential environmental constraints on the secure use of the TOE.

A.ADMIN	The TOE and the TOE environment are competently installed and administered. This includes that the TOE is running as dedicated machine with no other user applications or users and that the SecurID servers reside in the protected network.
A.AUTHKEY	Host keys used by the clients for server authentication are authentic. These keys are of high quality and the private key part used by the TOE is not being disclosed.
A.GATEWAY	The TOE is the only connection point between the unprotected and the protected network. All network traffic between the unprotected network and the protected network passes the TOE.
A.NOEVIL	The TOE administrators and the TOE environment administrators are trustworthy to perform discretionary actions in accordance with security policies and to not interfere with the underlying software, firmware or hardware.
A.PHYSICAL	The TOE is operated in a physically secure environment.
A.REMOTE	Remote users of the TOE originate from a well-managed user community. They follow the user guidance provided with the TOE and they protect their authentication credentials against unauthorized disclosure.
A.AUTH	In case of SecurID authentication is used, SecurID servers authenticate remote users, where the quality of the authentication credentials is not less than required for the password authentication.
A.TIME	The TOE environment provides a reliable time source.
A.RNG	The TOE environment provides good qualityrandom number generator.

3.2 Threats

This section describes the threat model for the TOE.

The primary **assets** that are the subject of attacks are the information and IT resources residing behind the AppGate Security Server in the protected network.

In order to attack these primary assets, the following assets must be subjects of attacks:

- data processed by the TOE (user data and / or TSF data)
- data transmitted between client and TOE (user data and / or TSF data)

The **threat agents** are categorized as either:

- unauthenticated individuals that are unknown to the TOE
- authenticated users of the TOE

Attackers in the position of authenticated users that are authorized for some services are assumed to try – either by accident or curiosity – to get access to resources they are not authorized to access.
An attacker is assumed to have limited resources and to come from a well-managed user

community in a non-hostile working environment. The TOE is not intended to be used in an environment in which the value of information and resources in the protected network is high, which implies that no protection against determined or sophisticated attacks is required.

Combining the assets and threat agents, the following threats need to be countered by the TOE:

T.BYPASS	An attacker is able to access protected network resources, user data or TSF data by circumventing the TOE Security Policy (TSP) enforcement functions by penetrating or manipulating portions of the TOE.
T.NETWORK	An attacker is able to gain access to user or TSF data by intercepting network sessions between remote clients and the TOE. This threat includes the disclosure of the sent data and/or the undetected modification of the captured network data.
T.AUTHENTIC	An attacker pretends to be another user, gaining unauthorized access to protected network resources, TSF data or user data.
T.AUTHORIZED	An attacker who is authenticated and may be authorized to have access to parts of information and/or resources in the protected network, gains access to protected network resources, TSF data or user data that he is not authorized to access.

3.3 Organizational Policies

P.ACCOUNT	Remote users shall be held accountable for their security-relevant actions.

4 Security Objectives

The security objectives provide a concise statement of the intended response to the security problem. It will describe which security needs will be addressed by the TOE and which will be addressed by the TOE environment, in the form of a statement of security objectives.

4.1 Security Objectives for the TOE

O.AUDIT	The TOE must provide accounting information for security-relevant actions of remote users, including authentication attempts, session establishment and configuration changes to the TOE. Administrators must be able to read the audit records. Administrators must be informed in case of storage problems.
O.AUTH	In case of password authentication, the TOE must uniquely identify and authenticate the claimed identity of remote users before they are allowed to access the data they are authorized to access. The TOE must also identify the user in case of SecurID authentication.
O.AUTHORIZE	The TOE must provide mechanisms to restrict the access of authenticated users to protected network resources, user data, and TSF data..
O.SECCOMM	The TOE must provide mechanisms to support server authenticity, integrity, and confidentiality of data that is transmitted between the clients and the TOE.

4.2 Security Objectives for the TOE Environment

OE.AUTHKEY	Host keys used by the clients for server authentication are authentic. These keys are of high quality and the private key part used by the TOE is not being disclosed .
OE.PHYSICAL	The TOE must be operated in a physically secure environment.
OE.NOEVIL	The TOE administrators and the TOE environment administrators must be trustworthy to perform discretionary actions in accordance with security policies and must not interfere with the underlying software, firmware or hardware.
OE.ADMIN	The TOE and the TOE environment are competently installed and administered. This includes that the TOE is running as dedicated machine with no other user applications or users and that the SecurID servers reside in the protected network.
OE.REMOTE	Remote users of the TOE originate from a well-managed user community. They follow the user guidance provided with the TOE and they protect their authentication credentials against unauthorized disclosure.
OE.GATEWAY	The TOE must be the only connection point between the unprotected and the protected network. All network traffic between the unprotected network and the protected network must pass the TOE.
OE.AUTH	In case of SecurID authentication is used, SecurID servers authenticate remote users.The quality of the authentication credentials is not less than required for the password authentication.
OE.TIME	The TOE environment must provide a reliable time source.
OE.RNG	The TOE environment must provide a good quality random number generator.

5 IT Security Requirements

The following table gives an overview of the functional components from the Common Criteria Part 2 that are relevant for this TOE.

Component	Component Name
FAU_GEN.1	Audit data generation
FAU_GEN.2	User identity association
FAU_SAR.1	Audit review
FAU_STG.3	Action in case of possible audit data loss
FAU_STG.4	Prevention of audit data loss
FCS_CKM.1	Cryptographic key generation
FCS_CKM.2	Cryptographic key distribution
FCS_COP.1	Cryptographic operation
FDP_ACC.1	Subset access control
FDP_ACF.1	Security attribute based access control
FIA_AFL.1	Authentication failure handling
FIA_ATD.1	User attribute definition
FIA_SOS.1	Verification of secrets
FIA_UAU.2	User authentication before any action
FIA_UAU.5	Multiple authentication mechanisms
FIA_UID.2	User identification before any action
FMT_MSA.1	Management of security attributes
FMT_MSA.3	Static attribute initialisation
FMT_SMF.1	Specification of Management Functions
FMT_SMR.1	Security Roles
FPT_RVM.1	Non-bypassability of the TSP
FTA_TSE.1	TOE session establishment
FTP_TRP.1	Trusted path

Table 1: Overview of the used SFRs of the TOE

5.1 TOE Security Functional Requirements

5.1.1 FAU_GEN.1 Audit data generation

FAU_GEN.1.1

The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions

b) All auditable events for the **not specified** level of audit; and

c) the following auditable events:

- **login attempts and logout**
- **actions performed by administrators**

- **password changed by user**

FAU_GEN.1.2

The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST the **severity level of event**.

Application note: The audit function is always active and cannot be started or shutdown as long as the TOE is running. The starting of the audit function is audited in the appgate logfile. The stopping of the audit function is audited in the syslog file (not part of the TOE). In addition, all actions that require logging are not performed by the TOE as long as the associated audit events cannot be recorded.
The subject identity is the session identity, which in turn is associated with the user identity at the beginning of the session.

5.1.2 FAU_GEN.2 User identity association

FAU_GEN.2.1

The TSF shall be able to associate each auditable event with the identity of the user that caused the event.

5.1.3 FAU_SAR.1 Audit review

FAU_SAR.1.1

The TSF shall provide **users with access rights to "Administration access" component** with the capability to read **date, time, event type, user identity, information of success/failure, severity level** from the audit records.

FAU_SAR.1.2

The TSF shall provide the audit records in a manner suitable for the user to interpret the information.

5.1.4 FAU_STG.3a Action in case of possible audit data loss

FAU_STG.3.1

The TSF shall **execute one of the following actions:**

- **issuing an SNMP trap**
- **send an e-mail to a specified address**
- **executing a specific command on the AppGate server**

if the audit trail exceeds **the administrator specified minimum free space.**

Application note: The administrator can specify both the minimum free space and the actions to be taken when this space is no longer available. The administrator can also specify arbitrary commands to be executed on the underlying operating system of the TOE or the client.

5.1.5 FAU_STG.3b Action in case of possible audit data loss

FAU_STG.3.1

The TSF shall **delete the oldest log files on a regular basis, until at least a specified amount of free space is available again** if the audit trail exceeds **the critical minimum free space of storage.**

Application note: Old compressed log files are kept until less than a minimum amount of kilobytes are left on the disk. Old compressed log files are then removed until enough disk space is free. The

minimum amount of free disk space is defined by the administrator. This threshold is different from the one defined in FAU.STG.3a. Note that log files that are too new will not be removed. The minimum age is specified by the administrator, but 7 days by default.

5.1.6 FAU_STG.4 Prevention of audit data loss

FAU_STG.4.1

The TSF shall **prevent auditable events, except those taken by the authorized user with special rights** and **no other action** if the audit trail is full.

Application note: The TOE prevents any remote user action that requires logging. Local administrative access is possible.

5.1.7 FCS_CKM.1 Cryptographic key generation (symmetric session keys)

FCS_CKM.1.1

The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm **as defined in the Secure Shell (SSH) Transport Layer Protocol [RFC 4253]** and specified cryptographic key sizes **192 bit (AES encryption key), 160 bit (HMAC authentication key)** that meet the following: **generation and exchange of session keys using Diffie-Hellman key exchange protocol as defined in [RFC 4253].**

Application note: Diffie-Hellman key exchange with SHA-1 is recommended by the SSH Transport Layer Protocol (standard SSHv2) that is used by the OpenSSH.

5.1.8 FCS_CKM.2 Cryptographic key distribution (symmetric session keys)

FCS_CKM.2.1

The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method **Diffie-Hellman key exchange** that meets the following: **distribution of keys as required by the Secure Shell Transport Layer Protocol [RFC 4253].**

Application note: The communication partners do not exchange the session keys themselves, they negotiate them with Diffie-Hellman key exchange. This SFR works in co-operation with FCS_CKM.1 to provide the key negotiation as required by [RFC 4253]. Both methods, diffie-hellman-group1-sha1 and diffie-hellman-group14-sha1, are supported.

5.1.9 FCS_COP.1a Cryptographic operation (encryption/decryption)

FCS_COP.1.1

The TSF shall perform **encryption and decryption** in accordance with a specified cryptographic algorithm **AES-CBC** and cryptographic key sizes **192 bits** that meet the following: **Advanced Encryption Standard (AES) [FIPS 197].**

5.1.10 FCS_COP.1b Cryptographic operation (keyed hashing generation and verification)

FCS_COP.1.1

The TSF shall perform **keyed hash generation and verification** in accordance with a specified cryptographic algorithm **HMAC-SHA1** and cryptographic key sizes **160 bit** that meet the following: **HMAC – Keyed-Hashing for Message Authentication**[**RFC2104].**

Application note: HMAC-SHA1 is recommended by SSHv2 standard SSH Transport Layer Protocol that is used by OpenSSH as means of integrity protection. The verifying side performs the verification through also calculating the keyed hash and compare it with the keyed hash value received from the other side.

5.1.11 FCS_COP.1c Cryptographic operation (signature generation)

FCS_COP.1.1

The TSF shall perform **digital signature generation** in accordance with a specified cryptographic algorithm **DSA** with cryptographic key sizes of **1024 bits** that meet the following:**Digital Signature Standard defined in [FIPS 186-2] and required by SSH Transport Layer Protocol [RFC 4253].**

5.1.12 FCS_COP.1d Cryptographic operation (digest generation and verification)

FCS_COP.1.1

The TSF shall perform **digest generation and verification** in accordance with a specified cryptographic algorithm **SHA-1** and **cryptographic key sizes: none** that meet the following: **US Secure Hash Algorithm 1 (SHA-1) [FIPS 180-2].**

Application note: The algorithm computes a hash-value of 160bit. The verifying side performs the verification through also calculating the hash and compare it with the hash value received from the other side.

5.1.13 FDP_ACC.1 Subset access control

FCS_ACC.1.1

The TSF shall enforce the **Role- and rule-based access SFP** on

subjects: user

objects: objects in the object-tree

and **operations:**

- **use the object according to its type**

Application note: The object-tree on which access control is applied contains role, folder and service objects. For more information, see section 2.4.2.1 in this document.

5.1.14 FDP_ACF.1 Security attribute based access control

FDP_ACF.1.1

The TSF shall enforce the **Role- and rule-based access SFP** to objects based on the following:

a) object attributes:

- **object name**
- **object type (role, folder, service)**
- **association to other objects. Possible associations are:**
  - **role to folder**
  - **role to service**
  - **folder to service**
  - **folder to folder**
- **access rules attached to associations of an object**
- **if the object is of type role, it can contain an access rule applicable to all users logging into that role**

b) user attributes

- **list of allowed roles**

c) **attributes of a session associated with a user:**

- **selected role**
- **client IP address**
- **client name (DNS)**
- **authentication method**
- **cryptographic method**

d) **manageable access rules. These are named boolean statements consisting of:**

- **comparisons based on the session attributes**
- **references to other existing access rules**
- **logical operators that connect these comparisons and references**

FDP_ACF.1.2

The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

An user assumes a role, i.e., gets associated with an object of type role according to the following rules:

- **the object is of type role**
- **the object is an element of the list of roles that the user can assume**
- **if an access rule exists for the role, the access rule grants access to the object**

For each type of access, the target object (other than role) must be accessible via the association path of the user's role.

For each type of access, all rules that exist along the association path from the user's role to the target object, must grant access.

If the target object is of type folder, the user can open it and reveal the containing folders or services.

If the target object is of type service, the user triggers the execution of the service that is represented by the service object.

If the target object is of type service and contains the special component "administration access", the user is allowed to administrate the TOE.

An access rule grants access if each comparison made on the user session attributes and the other nested access rules together with the boolean operators result to the result "true".

FDP_ACF.1.3

The TSF shall explicitly authorize access of subjects to objects based on the following additional rules: **none.**

FDP_ACF.1.4

The TSF shall explicitly deny access of subjects to objects based on the following additional rules: **none.**

Application note: Each role and association contains an access rule that default to the result "true".

5.1.15 FIA_AFL.1 Authentication failure handling

FIA_AFL.1.1

The TSF shall detect when **3** unsuccessful authentication attempts occur related to **the last 30 seconds.**

FIA_AFL.1.2

When the defined number of unsuccessful authentication attempts has been met or surpassed, the TSF shall **block any further login attempts from the same IP address within 30 seconds**.

Application note: Administrators can define how many failed login attempts are allowed within a configurable time frame before further login attempts from the same IP address are blocked. Any login attempt from the same IP address during the blocked period is counted as failed login attempt.

5.1.16 FIA_ATD.1 User attribute definition

FIA_ATD.1.1

The TSF shall maintain the following list of security attributes belonging to individual users:

- **user identifier**
- **expiration time of user account**
- **user roles**
- **allowed authentication method (Password/SecurID)**
- **applicable for users with password authentication:**
  - **password**
  - **user can change own password (yes/no)**
  - **user must change password on next login**
  - **user can change own password: always/never/after specific number of days after last change**
  - **warn user about password expiration: always/never/starting on a specific number of days before expiration**
  - **force user to change password at certain time intervals/never**
  - **expiration date of password or no disabling of expiration**

Application note: Administrators can specify if users defined in the user database are allowed to use passwords or SecurID for their authentication. Administrators may also specify as part of the TSF data management which SecurID server is used.

5.1.17 FIA_SOS.1 Verification of secrets (passwords)

FIA_SOS.1.1

The TSF shall provide a mechanism to verify that secrets meet:

- **minimum number of characters: 8**
- **minimum number of non-lowercase characters: 2**
- **minimum number of non-alphanumeric characters: 2**

Application note: A strength of function claim of SOF-medium applies to this SFR. This only applies for password authentication and not when using any external authentication mechanism.

5.1.18 FIA_UAU.2 User authentication before any action

FIA_UAU.2.1

The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application note: The communication of the authentication credentials as part of sshv2 is performed by the TOE according to [RFC 4252] for password authentication, and [RFC 4256] for SecurID authentication.

5.1.19 FIA_UAU.5 Multiple authentication mechanisms

FIA_UAU.5.1

The TSF shall provide:

- **UserID/Password-**
- **SecurID-**

based authentication mechanisms to support user authentication.

FIA_UAU.5.2

The TSF shall authenticate any user's claimed identity according to the **authentication method configured for each user by an authorized administrator**.

Application note: The password authentication is done as part of the TOE and the SecurID authentication is done by SecurID servers which is part of the IT environment. The administrator will configure one or both authentication methods to be used by the user. The user is only requested to authenticate once, so if multiple authentication methods are available to the user the user needs also to select the authentication method chosen in the beginning of the session. So either UserID/Password – or SecurID-based authentication is used.

5.1.20 FIA_UID.2 User identification before any action

FIA_UID.2.1

The TSF shall require each user to identify itself before allowing any other TSF-mediated actions on behalf of that user.

5.1.21 FMT_MSA.1 Management of security attributes

FMT_MSA.1.1

The TSF shall enforce the **Role- and rule-based SFP** to restrict the ability to **query, modify, delete, add** the security attributes:

- **user attributes**
- **password policy attributes**
- **access rules**
- **object attributes**
- **SecurID server connection configuration**

to **the administrator and the user (only in case of modifying the user attribute *password*)**.

Application note: The security attributes mentioned in this work unit are specified in more detail in the corresponding SFRs as follows:

- user attributes as defined by FIA_ATD.1 and FDP_ACF.1.1 b)
- password policy attributes as defined by FIA_SOS.1
- access rules as defined by FDP_ACF.1.1 d)
- object attributes as defined by FDP_ACF.1.1 a)

5.1.22 FMT_MSA.3 Static attribute initialisation

FMT_MSA.3.1

The TSF shall enforce the **Role-and rule-based access control SFP** to provide **restrictive** default values for security attributes that are used to enforce the SFP.

Application note: Restrictive default values are applicable to the list of allowed roles of users and associations between objects (identified by object name, object type). New users added to the TOE will not be able to access the TOE unless their password has been set by the administrator. The initial password given to them by the administrator must comply with the password policy just as passwords that are later selected by the users.

FMT_MSA.3.2

The TSF shall allow the **administrator** to specify alternative initial values to override the default values when an object or information is created.

Application note: The specification of alternative initial values is limited to the associations between objects (together with object name and object type). The administrator may specify which combinations of folders and services a user is allowed to access.

5.1.23 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

The TSF shall be capable of performing the following security management functions:

- **audit event management**
- **user management and management of default user settings**
- **authentication data management (only for password authentication)**
- **access control management**
- **SecurID server connection configuration**

Application note: While the administrative functions are available to administrators only, parts of the the authentication data management may be given to any user, allowing them to change their own passwords.

5.1.24 FMT_SMR.1 Security Roles

FMT_SMR.1.1

The TSF shall maintain the roles **user and administrator.**

FMT_SMR.1.2

The TSF shall be able to associate users with roles.

Application note: In this context, user and administrative roles are meant as security roles. A user has the security role “administrator” if he has access to the "Administration access" component. If not, he is in the position of a normal user. In addition, the administrator needs to have network access to the TOE, which is defined with the help of an IP access component that is assigned to an administrative user. This component allows traffic between source and destination IP addresses, but this is not considered a security function.

5.1.25 FPT_RVM.1 Non-bypassability of the TSP

FPT_RVM.1.1

The TSF shall ensure that TSP enforcement functions are invoked and succeed before each function within the TSC is allowed to proceed.

5.1.26 FTA_TSE.1 TOE session establishment

FTA_TSE.1.1

The TSF shall be able to deny session establishment based on

- **user identifier, password, SecurID authentication data used in the TOE environment (identification and authentication fails)**
- **date of user account (user account is expired)**

Application note: Security attributes are specified that may cause the deny of a session establishment. Related to these attributes, the conditions for a deny is provided in brackets. As a result of the session establishment, the user is associated with a role and other session attributes as defined by FDP_ACF.1.

5.1.27 FTP_TRP.1 Trusted path

FTP_TRP.1.1

The TSF shall provide a communication path between itself and **remote** users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from modification or disclosure.

FTP_TRP.1.2

The TSF shall permit **remote users** to initiate communication via the trusted path.

FTP_TRP.1.3

The TSF shall require the use of the trusted path for **all services**.

Application note: Remote users are both normal users and administrative users. Note that the clients being part of this trusted path are part of the TOE environment.

5.2 TOE Security Assurance Requirements

The TOE assurance requirements are incorporated by reference to Evaluation Assurance Level (EAL) 2 augmented with ALC_FLR.1 as specified in Part 3 of the CC. The following table provides an overview of the assurance components.

Assurance class	Assurance components
Configuration management	ACM_CAP.2 Configuration items
Delivery and operation	ADO_DEL.1 Delivery procedures
	ADO_IGS.1 Installation, generation, and start-up procedures
Development	ADV_FSP.1 Informal functional specification
	ADV_HLD.1 Descriptive high-level design
	ADV_RCR.1 Informal correspondence demonstration
Guidance and Documentation	AGD_ADM.1 Administrator guidance
	AGD_USR.1 User guidance
Life cycle support	ALC_FLR.1 Basic flaw remediation
Tests	ATE_COV.1 Evidence of coverage
	ATE_FUN.1 Functional testing
	ATE_IND.2 Independent testing - sample
Vulnerability assessment	AVA_SOF.1 Strength of TOE security function evaluation
	AVA_VLA.1 Independent vulnerability analysis

Table 2: Overview of the used Security Assurance Requirements

5.3 Security Functional Requirements for the IT Environment

The TOE relies on its environment to enforce certain security functionality. The requirements to operate the TOE securely in a reliable environment are listed below.

Component	Component Name
FIA_ATD.1	User attribute definition
FIA_UAU.2	User authentication before any action
FIA_UAU.4	Single-use authentication mechanisms
FIA_UID.2	User identification before any action
FPT_STM.1	Reliable time stamps
FCS_RNG.1	Generation of random numbers

Table 3: Overview of the used SFRs for the IT environment

5.3.1 FIA_UAU.2 User authentication before any action

FIA_UAU.2.1

The **security functions of the TOE environment** shall require each user to be successfully authenticated before allowing any other **actions mediated through the TOE environment** on behalf of that user.

Application note: This SFR was refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

Application note: The communication of SecurID authentication credentials as part of sshv2 is performed by the TOE according to [RFC 4256]. The authentication process itself is part of the TOE environment.

5.3.2 FIA_UID.2 User identification before any action

FIA_UID.2.1

The s**ecurity functions of the TOE environment** shall require each user to identify itself before allowing any other **actions mediated through the TOE environment** on behalf of that user.

Application note: This SFR was refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

5.3.3 FIA_UAU.4 Single-use authentication mechanisms

FIA_UAU.4.1

The s**ecurity functions of the TOE environment** shall prevent reuse of authentication data related to **the authentication in the TOE environment**.

Application note: This SFR was refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

5.3.4 FIA_ATD.1 User attribute definition

FIA_ATD.1.1

The s**ecurity functions of the TOE environment** shall maintain the following list of security attributes belonging to individual users:

- **user identity**
- **personal identification number**
- **authentication code of the SecurID token**

Application note: For user being authenticated using SecurID, the user information is also part of the TOE environment. The user identity defined on the TOE must match the user identity on the SecurID server (TOE environment). This SFR was refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

5.3.5 FPT_STM.1 Reliable time stamps

FPT_STM.1.1

The **security functions of the TOE environment** shall be able to provide reliable time stamps **used by the TOE.**

Application note: This SFR was refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

5.3.6 FCS_RNG.1 Generation of random numbers

FCS_RNG.1.1.

The **security functions of the TOE environment** shall provide a **non-physical true** random number generator that implements **an estimate of the entropy of the random numbers.**

FCS_RNG.1.2

The **security functions of the TOE environment** shall provide random numbers that meet the requirement **of allowing to extract random numbers only if the estimate of the entropy exceeds a defined threshold.**

Application note: This SFR is not part of CC Part2, but is an explicitly stated SRFdrawn from the certified Protection Profile [SPP]. This SFR was further refined to avoid misunderstandings between TOE/TSFs, and TOE environment systems and their security functions.

6 TOE Summary Specification

The TOE summary specification provides a complete high-level definition of the security functions and assurance measures of the TOE and their relationship to the security functional and assurance requirements of this ST.

6.1 TOE Security Functions

6.1.1 SF.AU – Auditing

The TOE provides the generation of audit records for security-relevant events.

SF.AU.1

The following events are logged:

- login attempts and logout
- actions performed by administrators
- password change by user

For every event, the following information is logged:

- date
- time
- type of event
- session identity of the event (sessions are associated with users as part of the user authentication)
- explicit information when something failed
- a severity level is assigned:
  - **info**: Informational messages, logs normal use of the AppGate server. Examples: user logins and starting/stopping of IP accesses.
  - **warning**: Non-systematic errors, rather "random" events. Examples: erroneous packets received, the AppGate hits a soft limit, such as the maximum number of connections.
  - **error**: Systematic errors. Example: a syntax error while parsing a file, such as sdb.db.
  - **CRITICAL**: Critical errors, should be acted upon immediately. Examples: AppGate could not create a PID file or read a configuration file.
  - **SUSPECT**: A suspected attempt at a security violation. Examples: a user failed to log on to the AppGate server, or the server receives a different host name when doing a reverse DNS lookup.

The events mentioned above are all stored to the file *appgate.log*.

The current log file is saved and renamed with the date every 24 hours, and a new log file will be created to be used the next 24 hours.

SF.AU.2

If the TOE cannot generate audit records as described in SF.AU.1, the TOE will refuse all user action that require generating audit records, e.g., new connections from remote users. Local administrative access to the TOE using the TOE environment is still possible.

SF.AU.3

The system can create notifications (SNMP trap, e-Mail to specified address, command execution on the AppGate Server) if the storage exceeds a pre-defined limit. Different limits can be set, separating the notifiable storage status in:

- high critical: X MB left
- high warning: X MB left
- normal: no value (the normal status exists if none of the other thresholds apply)
- low warning: X MB left
- low critical: X MB left (1000MB recommended),

where the threshold X can be configured by the administrator. However, there are only reasonable values for X above 100MB. A threshold below 100 may not have any effect because of SF.AU.4.

SF.AU.4

If less than 100MB of the storage is free, the oldest log files are deleted until at least 100MB of storage on the audit trail is free again, provided that these log files are older than 7 days. This is done every (1) hour.

SF.AU.5

The audit records can be read by administrators using the Log GUI within the AppGate Console. The Log GUI assists the administrator in creating graphical statistics about the log entry information and allows him to specify parameters for reviewing the audit data selectively.

6.1.2 SF.IA – Identification and Authentication

Identification of user accounts is performed by checking existing user accounts in the local database. After the existence of the user identity is verified, identification is finished and authentication starts. The TOE authenticates users through passwords stored in the local database or requests the SecurID server to authenticate the user. The user name and the authentication credentials are transmitted embedded within the SSHv2 communication as defined in the the SSH User Authentication Protocol. SF.IA also covers some additional rules for the session establishment between a remote user and the TOE.

6.1.2.1 User identification

SF.IA.1

Each user is required to provide an identification string (user identifier) that the TOE compares with his information in its local database to find out whether the user has an account.

SF.IA.2

If the identification attempt fails, the user receives an error message stating that the user name or credentials are wrong, independent of the reason of identification and authentication error.

6.1.2.2 User authentication

SF.IA.3

After the user selected an authentication method, the TOE verifies whether the user is allowed to use the selected authentication method. Two authentication methods are part of the evaluated configuration. They are password authentication and the SecurID.

SF.IA.4

Each user is required to provide authentication credentials. If the selected authentication method is password authentication, the authentication is performed according to [RFC 4252], and the TOE compares the provided information with the information stored in the local database (contains password hashes). If the password matches the user password in the local database of the TOE, the authentication is successful. If the selected authentication method is SecurID, the authentication is performed according to [RFC 4256], and the request is sent to the SecurID server which decides if the authentication is successful or not.

SF.IA.5

For password authentication user passwords must comply to the password policy enforced by the TOE as follows:

- minimum number of characters: 8
- minimum number of non-lowercase characters: 2
- minimum number number of non-alphanumeric characters: 2

This applies to remote users that change their password during SSH authentication.

Based on the password quality enforced by this password policy, a strength of function claim SOF-medium is made for the password authentication mechanism.

SF.IA.6

If the authentication fails, the user is prompted to enter the user name and credentials again. The user also receives an error message as described in SF.IA.2.

SF.IA.7

If the password authentication is used, and the provided password is correct, but the password is expired, the authentication attempt fails and an error message is displayed.

SF.IA.8

If the authentication method is SecurID authentication, the TOE passes the user name and authentication credentials to the SecurID server requesting the SecurID server to authenticate the user.

SF.IA.9

The user can be forced to change his password during login, or he can be warned that his password is going to expire (after successful authentication).

SF.IA.10

After successful authentication, the information of the user is associated with user identification string that uniquely identifies the user.

6.1.2.3 Multiple failed login attempts and deny of session establishment

SF.IA.11

Administrators can define how many failed login attempts are allowed within a given time frame before further login attempts from the same IP address are blocked. Any login attempt from the same IP address during the blocked period is counted as failed login attempt. Required settings are 3 allowed authentication attempts within 30 seconds.

SF.IA.12

After a successful authentication, if the user account is expired the session between the remote user and the TOE is not established.

6.1.3 SF.AC - Access control

Access to resources in the protected network or administrative access to the TOE is controlled by the TOE. This is done through associating roles and access rules. Roles are used in a static way as the administrator defines the allowed roles for each user and also defines for each role to which folders and services this role grants access. On the other hand, access rules are a dynamic way of deciding on access to an object, because attributes are checked during the user establishes a session to the TOE and requests access to an object.

6.1.3.1 Roles, folders, services and components

Roles, folders, services and components are objects on which access is controlled, either by the association between these objects, or by access rules (see SF.AC.3-4).

SF.AC.1

A user is configured to have access to none, one or more roles. The user can have only one role per session. If a user is not allowed to have at least one role, the user is not granted any access to the TOE (other than the authentication interface) or the protected resources. This happens if the user is not associated to any role, or if access rules deny access to any role, defined for the user.

SF.AC.2

A role object can contain direct or indirect associations to other objects: folders, services and components (always implicit via a service).

A folder is used to group services or other folders (sub-folders). The user can use the service through opening it and reveal the contained objects.

A service is a collection of components, where each component performs a single task, e.g., defines the IP address for the target resource of a service. The access to a service allows the user to actually use the network service that this object represents.

Roles, folders, services and component objects build a tree, with the role node as its top. An user can access all the objects in this tree, unless restricted by access rules.

6.1.3.2 Access rules

SF.AC.3

Access rules control access to roles and to associations of role/folders, role/services, folder/folders and folder/services. Access rules do not control associations between services and components. A user that has access to a specific service also immediately has access to the components of that the service consist of.

SF.AC.4

Access rules perform access decisions through regular expressions on user session attributes. Such attribute checks are bound together with boolean operators to form the overall result “true” or “false”. In case of “true”, access is granted, in case of “false”, access is denied. The security-relevant attributes are:

- client IP address
- client name (DNS)
- authentication method
- cryptographic methods
- references to other existing access rules that should be evaluated
- logical operators (and, or, not)

Rules can also consist of only “true” or “false”. In addition, the overall result can be negated.

To be able to access a target object, all access rules of the associations, beginning at the role and propagating to the target object in the object-tree, must grant access.

All roles and associations between objects are connected to rules. If these rules are unspecified, they grant access by default.

Administration access

SF.AC.5

If one of the accessed services contains the "Administration access" component, the user is allowed to manage the TOE through the AppGate Console. This means that the user assumes the security role “administrator”.

6.1.4 SF.CRYPTO – Cryptographic functions

The communication between user and the TOE is secured as specified in the SSH Transport Protocol [RFC 4253] and is not described in more detail here. Please refer to section 2.4 of this document for an overview the secure communication and the used cryptographic mechanisms. The cryptographic actions are all part of the standard SSH protocol as specified as part of [RFC 4253], starting with the server authentication as described in SF.CRYPTO.2, followed by the Diffie-Hellman key exchange as described in SF.CRYPTO.1. After server authentication and key exchange succeeded, encryption/ decryption and the generation/verification of the message authentication codes is performed as described by SF.CRYPTO.3 and SF.CRYPTO.4.

SF.CRYPTO.1

The SSH protocol specified in [RFC 4253] is relying on Diffie-Hellman key exchange protocol used to provide a shared secret that cannot be determined by either party alone. The symmetric session key for encryption (AES, 192bit) as well as the key for the message authentication (HMAC-SHA-1, 160-bit key length) are derived from the shared secret and the exchanged hash on each side, where only the key exchange operations of the TOE are of interest.

SF.CRYPTO.2

For server authentication, the SSH protocol specified in [RFC 4253] is relying on the generation and verification of digital signatures according to the Digital Signature Standard [FIPS 186-2] using the private server key with 1024 bit key length. SHA-1 [FIPS 180-2] is used to compress the data input for generation of the signature.

SF.CRYPTO.3

The SSH protocol specified in [RFC 4253] is relying on the encryptionand decryption being performed through the symmetric algorithm AES in CBC-mode with 192-bit key length [FIPS 197].

The encryption is done for all outgoing data that is sent to the SSH clients in the unprotected network. The decryption is done for all incoming data that comes from the SSH clients in the unprotected network.

SF.CRYPTO.4

The SSH protocol specified in [RFC 4253] is using a message authentication code (according HMAC-SHA-1 [RFC 2104]) using the respective authentication key (160-bit length) and a cryptographic hash function is calculated over the payload of the transmitted packets, providing the ability to detect loss of integrity and authenticity on the receiving side. SHA-1 [FIPS 180-2] is used as base for generation of the keyed hash.

The message authentication code is calculated and sent to the SSH clients for all outgoing data that is sent to the SSH clients in the unprotected network. The message authentication code is calculated and verified for all incoming data that comes from the SSH clients in the unprotected network

6.1.5 SF.MGMT – Security management

Security functions can be managed by an administrator as follows:

- auditing (SF.AU)
- identification and authentication (SF.IA)
- authorization (SF.AC)

In order to manage these security functions, the administrator manages the following security attributes:

- password policy (SF.IA)
- user attributes (SF.IA, SF.AC)
- access rule attributes (SF.AC)
- associations between roles/folders/services /components (SF.AC)

Users may be allowed to change their own passwords. All other security management functions are performed by administrators.

6.1.5.1 Audit management

SF.MGMT.1

Different limits can be set, separating the notifiable storage status in:

- high critical: X MB left
- high warning: X MB left
- normal: no value (the normal status exists if none of the other thresholds apply)
- low warning: X MB left
- low critical: X MB left (1000MB recommended),

where the threshold X can be configured by the administrator.

SF.MGMT.2

The following notifications can be specified if the limit of free storage is exceeded:

- SNMP traps
- e-mail
- execution of commands on the AppGate Server.

6.1.5.2 Password authentication management

SF.MGMT.3

The following password quality attributes can be set:

- minimum number of characters (should be at least 8)
- minimum number of non-lowercase characters (should be at least 2)
- minimum number of non-alphanumeric characters (should be at least 2)

SF.MGMT.4

Additional configuration can be managed through a password policy, including:

- number of days after the last password change until the user is warned to change his password again
- number of days after the last password change until the user is forced to change his password again
- number of days after the last password change until the password is disabled

6.1.5.3 SecurID authentication management

SF.MGMT.5

A configuration file can be uploaded by an administrator which contains information for communicating with an SecurID server.

6.1.5.4 User accounts

SF.MGMT.6

The following user attributes can be configured for each user:

- user identity
- expiration time of user account
- permitted user roles
- allowed authentication method (Password/SecurID)
- password

SF.MGMT.7

When password authentication is used, the default user settingcan be overridden (this does not apply to password quality attributes), which additionally adds the following information to the account of that user:

- user can change own password (yes | no)
- user must change password on next login
- user can change own password always/never/after specific number of days after last change
- warn user about password expiration: always/never/starting on a specific number of days before expiration
- force user to change password at certain time intervals/never
- expiration date of password or no disabling of expiration

SF.MGMT.8

Only passwords that conform to the password policy are accepted by the TOE. This applies to passwords that are modified by remote users during the SSH authentication, and to passwords created/modified by administrators using the AppGate Console.

SF.MGMT.9

The user can change his own password if the administrator configured his account to do so.

6.1.5.5 Access rule management

SF.MGMT.10

The access rules consist of the following manageable data:

- name of access rule
- from (specifies IP addresses or host names, also using wild cards)
- authentication method allowed
- cryptographic methods
- another access rule that is checked before
- AND, OR, NOT

The values specified in an access rule (except its name) are compared with actual parameters of a user session. Each such check evaluates to “true” or “false”. Checks can be be logically bound together with AND, OR or NOT. An example access rule would be:

(not from(*.appgate.com)) and auth(certificate)

SF.MGMT.11

Access rules can be attached between:

- a role and a folder
- a role and a service
- a folder and a folder
- a folder and a service

Access rules can also be attached directly to a role.

6.1.5.6 Role management

SF.MGMT.12

The following security-relevant parts of a role can be configured:

- name
- access rule
- list of folders/services that belong to the role

6.1.5.7 Folder, service and component management

SF.MGMT.13

Named folders, services and components can be created and associated to roles, folders services as follows:

- folder to role
- service to role
- folder to folder
- service to folder
- component to service

6.2 TOE Assurance Measures

This chapter gives information about the measures the developer has taken to achieve the desired EAL 2 augmented assurance level. Due to the fact that the TOE security assurance requirements are exclusively based on ISO/IEC 15408 assurance components, we only provide a reference to the documents that show that the assurance requirements are met.

SAR	Documentation describing how the requirements are met
ACM_CAP.2	The configuration management tool CVS is used to manage the configuration items of the TOE. The manual of the CVS tool and the procedures for using it are documented in separate documents. The TOE is referenced by unique version numbers and is labeled with its reference. ACM documentation is provided . The CVS is also used used to provide automated support for generating the TOE from its implementation representation.
ADO_DEL.1	Description of processes used when delivering the TOE to the user's site are provided as part of the delivery documentation.
ADO_IGS.1	Installation generation and start-up is not done by the end user. The TOE is provided as an appliance to the end user ready to be used. Instructions how to connect and maintain the TOE are provided with the administrator guidance.
ADV_FSP.1	Documentation of the security functions and their interfaces, including explanations of their behavior concerning error messages, exceptions and details of effects is provided as part of the functional specification. Also showing that the security functions are sufficient to cover the security functional requirements.
ADV_HLD.1	Documentation on the high-level design, outlining the major structural units of the TOE and the security functionality associated with them is provided as part of the high-level design.
ADV_RCR.1	Documentation of correspondence between the TOE summary specification and the functional specification, and between the functional specification and the high-level design to ensure the correct and consistent instantiations of the TOE representations.
AGD_ADM.1	Documentation to be used by those persons responsible for configuring, maintaining, and administering the TOE in a manner consisting with the evaluated configuration is provided as part of the AppGate guide.
AGD_USR.1	Documentation to be used by non-administrative users of the TOE is provided as part of the AppGate guide.
ALC_FLR.1	Documentation about processes and policies to track and correct security flaws discovered by TOE consumers.
ATE_COV.1	An analysis of the test coverage is provided in the AppGate test coverage documentation document.
ATE_FUN.1	Testing will be performed on the platform as defined by the ST. Test results are documented such that the testing can be repeated. Evidence showing that the TOE exhibits the properties necessary to satisfy the functional requirements of its ST.
ATE_IND.2	Independent testing will be performed by the evaluation facility. The TOE and an equivalent set of resources are provided to the evaluation facility in a manner suitable for testing.
AVA_SOF.1	The TOE includes one mechanism having a strength of TOE security function claim, the password authentication. For this mechanism, a strength of TOE security function analysis is provided.
AVA_VLA.1	Analysis for identifying obvious vulnerabilities and confirmation that they cannot be exploited in the given environment is provided as part of

SAR	Documentation describing how the requirements are met
	the vulnerability analysis

Table 4: TOE Assurance Measures

7 PP Claims

This Security Target does not claim compliance with any Protection Profile.

8 Rationale

The rationale section demonstrates how the security objectives of the TOE are met and how objectives, threats and security functions relate to each other. The rationale section will identify, which security functions contribute to which objectives and which threats are countered by the individual security functions.

8.1 Security Objectives Rationale

8.1.1 Security Objectives Coverage

The following tables provide a mapping of security objectives to the environment defined by the threats, policies and assumptions, illustrating that each security objective covers at least one threat and that each threat is countered by at least one objective, assumption or policy.

Security objective	Corresponding threat, assumption or policy
O.AUDIT	P.ACCOUNT
O.AUTH	T.AUTHENTIC, P.ACCOUNT, T.AUTHORIZED
O.AUTHORIZE	T.BYPASS, T.AUTHORIZED
O.SECCOMM	T.NETWORK

Table 5: TOE objectives mapped to threats, and OSPs

Security objective environment	Corresponding threat, assumption or policy
OE.ADMIN	A.ADMIN
OE.AUTHKEY	A.AUTHKEY, T.NETWORK
OE.GATEWAY	A.GATEWAY
OE.NOEVIL	A.NOEVIL
OE.PHYSICAL	A.PHYSICAL
OE.REMOTE	A.REMOTE
OE.TIME	A.TIME, P.ACCOUNT
OE.AUTH	A.AUTH, T.AUTHORIZED, T.AUTHENTIC
OE.RNG	A.RNG, T.NETWORK

Table 6: TOE environment objectives mapped to threats, assumptions and OSPs

8.1.2 Security Objectives Sufficiency

The following rationale provides justification that the security objectives are suitable to counter each individual threat and that each security objective tracing back to a threat, when achieved, actually contributes to the removal, diminishing or mitigation of that threat:

Threat	Rationale for Objectives
T.BYPASS	The threat of an attacker to bypass TSFs is mitigated by enforcing controlled access to protected resources. This is ensured by *O.AUTHORIZE*.
T.NETWORK	Disclosure of information of user network traffic or its undetected modification is prevented by *O.SECCOMM* that enforces server authenticity (prevents man-in-the-middle attacks), encryption and the integrity of data through the use of SSHv2. Server authenticity is further supported by OE.AUTHKEY. The quality of the encryption keys and the message authentication keys used within Diffie-Hellman (DH parameters) are ensured by using random numbers of good quality when computing such keys, thus is ensured by using a secure random number generator as required in OE:RNG.
T.AUTHENTIC	*O.AUTH* ensures that all users are properly identified. *O.AUTH* also ensures authentication of users in case of password authentication. This is further supported by *OE.AUTH*, which requires that, in case of SecurID authentication, SecurID servers authenticate the remote users.
T.AUTHORIZED	Authorization is grounded on *O.AUTH* that verifies users' identities. According to a user identity, individual access to defined objects is granted or denied, as defined by *O.AUTHORIZE*. This is further supported by *OE.AUTH* which requires, that in case of SecurID authentication, SecurID servers must authenticate remote users, which is subsequently used to grant or to deny access to resources.

Table 7: Sufficiency of objectives countering threats

The following rationale provides justification that the security objectives for the environment are suitable to cover each individual assumption, that each security objective for the environment that traces back to an assumption about the environment of use of the TOE, when achieved, actually contributes to the environment achieving consistency with the assumption, and that if all security objectives for the environment that trace back to an assumption are achieved, the intended usage is supported:

Threat	Rationale for Objectives
A.PHYSICAL	*OE.PHYSICAL* requires that the TOE is operated in a physical secure environment.
A.NOEVIL	OE.NOEVIL requires that the TOE administrators and the TOE environment administrators are trustworthy to perform discretionary actions in accordance with security policies and that they in no way interfere with the underlying software, firmware or hardware.
A.ADMIN	*OE.ADMIN* requires that the TOE and the TOE environment are competently installed and administered. OE.ADMIN also requires that the TOE is running as a dedicated machine with no other user applications or users. *OE.ADMIN* especially requires that the SecurID servers reside in the protected network.
A.REMOTE	*OE.REMOTE* requires that remote users of the TOE originate from a well-managed user community. It also requires that they follow the user guidance provided with the TOE and it requires that they protect their authentication credentials against unauthorized disclosure.
A.GATEWAY	*OE.GATEWAY* requires that the TOE is the only connection point between the unprotected and the protected network. All network traffic between the unprotected network and the protected network shall pass the TOE.
A.AUTH	*OE.AUTH* requires that SecurID servers authenticate remote users. *OE.AUTH* also requires that the quality of the authentication credentials is not less than required for the password authentication..
A.TIME	*OE.TIME* requires a reliable time stamp.
A.AUTHKEY	*OE.AUTHKEY* requires that host keys used for the server authentication are authentic. *OE.AUTHKEY* requires that these keys are of high quality and that the private key part held by the TOE is not being disclosed.
A.RNG	OE.RNG requires that a good quality random number generator is provided.

Table 8: Sufficiency of objectives meeting assumptions

The following table provides justification that each OSP is covered by at least one security objective and explains how the objectives contribute to cover the OSPs:

Threat	Rationale for Objectives
P.ACCOUNT	*O.AUDIT* requires to record any security-relevant action by a remote user in a secure way in order to hold remote users responsible for their actions. The association between user subjects and user identities is supported by *O.AUTH. O.AUDIT* also requires that administrators can read the audit records, and administrators must be informed in case of storage problems. The correct time stamp of a logged event is ensured by *OE.TIME*.

Table 9: Sufficiency of objectives meeting OSPs

8.2 Security Requirements Rationale

8.2.1 Security Functional Requirements Coverage

The following table demonstrates how each TOE security functional requirement can be traced back to at least one security objective for the TOE:

SFR	is necessitated by
FAU_GEN.1	O.AUDIT
FAU_GEN.2	O.AUDIT
FAU_SAR.1	O.AUDIT
FAU_STG.3a	O.AUDIT
FAU_STG.3b	O.AUDIT
FAU_STG.4	O.AUDIT
FCS_CKM.1	O.SECCOMM
FCS_CKM.2	O.SECCOMM
FCS_COP.1a	O.SECCOMM
FCS_COP.1b	O.SECCOMM
FCS_COP.1c	O.SECCOMM
FCS_COP.1d	O.SECCOMM
FDP_ACC.1	O.AUTHORIZE
FDP_ACF.1	O.AUTHORIZE
FIA_AFL.1	O.AUTH
FIA_ATD.1	O.AUDIT, O.AUTH, O.AUTHORIZE
FIA_SOS.1	O.AUTH
FIA_UAU.2	O.AUTH
FIA_UAU.5	O.AUTH
FIA_UID.2	O.AUTH
FMT_MSA.1	O.AUTHORIZE
FMT_MSA.3	O.AUTHORIZE
FMT_SMF.1	O.AUTH, O.AUTHORIZE, O.AUDIT
FMT_SMR.1	O.AUTHORIZE
FPT_RVM.1	O.AUTHORIZE
FTP_TRP.1	O.SECCOMM
FTA_TSE.1	O.AUTH

Table 10: Mapping TOE SFRs to objectives

The following table demonstrates how each TOE environment security functional requirement can be traced back to at least one security objective for the TOE environment:

SFR	is necessitated by
FIA_ATD.1	OE.AUTH
FIA_UAU.2	OE.AUTH
FIA_UAU.4	OE.AUTH
FIA_UID.2	OE.AUTH
FPT_STM.1	OE.TIME
FCS_RNG.1	OE.RNG

Table 11: Mapping of TOE environment SFRs to environment objectives

8.2.2 Security Functional Requirements Sufficiency

The following list provides justification for each security objective for the TOE, showing that the TOE security functional requirements are suitable to meet and achieve the security objectives:

O.AUDIT — FAU_GEN.1 and FAU_GEN.2 require that relevant audit data is stored in the audit trail. Authorized users are allowed to read the record data as specified by FAU_SAR.1. The already-written data is preserved also in the case that the audit trail becomes full as required by FAU_STG.4. Reasonable early notification to the administrator is required by FAU_STG.3a. FAU_STG.3b requires that the TOE frees storage if critical storage limits are exceeded.
Furthermore, FIA_ATD.1 requires that a user can be held accountable for his actions with the help of the audit records. This requires appropriate user attributes (FIA_ATD.1). Finally, FMT_SMF.1 allows to manage the audit/logging security functionality.

O.AUTH — Correctly handling password and user name information is required by FIA_ATD.1. Protecting passwords from brute force and dictionary attacks in order to secure the strength of the authentication method is specified by FIA_SOS.1 and supported by FIA_AFL.1. FIA_UAU.2 and FIA_UID.2 require that a user must strictly satisfy identification and authentication before any other actions on behalf of that user is allowed. FIA_UID.2 also requires user identification in case of SecurID authentication. FIA_UAU.5 specifies the use of multiple authentication methods. FTA_TSE.1 supports O.AUTH with rules for denying session establishments from a remote user to the TOE. Finally, FMT_SMF.1 allows to manage the authentication security functionality.

O.AUTHORIZE — Authorized access to resources is controlled by a Role- and rule-based access control SFP as specified by FDP_ACF.1 and FDP_ACC.1. Authorization for access is verified and enforced for each user on every new successful login on all relevant actions in respect to the SFP are validated (FPT_RVM.1). Domain separation (FPT_SEP.1) for supporting reference mediation (FPT_RVM.1) is not required, because the TOE runs on a dedicated machine (single-purpose, supported by A.ADMIN), which means that the TSF run within a single domain under complete control of the TOE. FIA_ATD.1 requires the security attributes to be correctly associated to a user. The ability of the TOE to distinguish between normal and administrative users is specified by FMT_SMR.1. The security attributes associated can be managed by authorized users as specified by FMT_MSA.1/FMT_SMF.1. The restrictive default values related to the access control policy are specified by FMT_MSA.3.

O.SECCOMM	A secure communication path is specified by FTP_TRP.1. Information on this path should be trusted in terms of confidentiality and integrity, which is supported by the requirements for secure key generation (FCS_CKM.1) and key distribution (FCS_CKM.2). These two SFRs are both used to model the requirements for the Diffie-Hellman key exchange mechanism. O.SECCOMM is further met through cryptographic operations encryption/decryption (FCS_COP.1a) and keyed hash generation/verification (FCS_COP.1b). Keyed hashes again require a standardized method of hash generation and verification, which is specified by FCS_COP.1d. Use of host keys for server authentication using digital signatures in order to prevent man-in-the-middle attacks that may corrupt the secure communication is specified by FCS_COP.1c. FCS_COP.1c also relies on the hashing of data for the signature generation process, using FCS_COP.1d.

8.2.3 Sufficiency of the Security Requirements of the TOE environment

The following list provides justification for each security objective for the TOE environment, showing that the security functional requirements for the TOE environment are suitable to meet and achieve the security objectives of the TOE environment:

OE.TIME	This objective is met by FPT_STM.1(e), which requires a secure time source in the environment.
OE.PHYSICAL	No dependencies to any SFR.
OE.NOEVIL	No dependencies to any SFR.
OE.ADMIN	No dependencies to any SFR.
OE.REMOTE	No dependencies to any SFR.
OE.AUTH	FIA_UID.2(e) and FIA_UAU.2(e) require that the SecurID server in the IT environment authenticate the remote users. The authentication is detailed by FIA_UAU.4(e) that requires that the SecurID authentication credentials are unique and cannot be reused. The user attributes on which the identification and authentication works is specified by FIA_ATD.1(e).
OE.GATEWAY	No dependencies to any SFR.
OE.AUTHKEY	No dependencies to any SFR.
OE.RNG	This objective is met by FCS_RNG.1(e), which requires a good quality random number generator.

8.2.4 Security Requirements Dependency Analysis

Following the Common Criteria and choosing security requirements to be met by a TOE, certain dependencies on other security requirements may arise. The following section shows whether these dependencies are resolved and, in case they are not, gives reasons. SFRs on the TOE environment are marked with an additional "(e)". If there are alternative requirements to resolve a dependency, the valid requirements are put in **bold** letters.

SFR	Dependencies	Resolution
FAU_GEN.1	FPT_STM.1	FPT_STM.1 (e)
FAU_GEN.2	FAU_GEN.1 FIA_UID.1	FAU_GEN.1 FIA_UID.2
FAU_SAR.1	FAU_GEN.1	FAU_GEN.1

SFR	Dependencies	Resolution
FAU_STG.3a	FAU_STG.1	-unresolved-
FAU_STG.3b	FAU_STG.1	-unresolved-
FAU_STG.4	FAU_STG.1	-unresolved-
FCS_CKM.1	[**FCS_CKM.2**; FCS_COP.1]	FCS_CKM.2
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FCS_CKM.2	[FDP_ITC.1; FDP_ITC.2; **FCS_CKM.1**]	FCS_CKM.1
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FCS_COP.1a	[FDP_ITC.1; FDP_ITC.2; **FCS_CKM.1**]	FCS_CKM.1
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FCS_COP.1b	[FDP_ITC.1; FDP_ITC.2; **FCS_CKM.1**]	FCS_CKM.1
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FCS_COP.1c	[FDP_ITC.1; FDP_ITC.2; FCS_CKM.1]	-unresolved-
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FCS_COP.1d	[FDP_ITC.1; FDP_ITC.2; FCS_CKM.1]	-unresolved-
	FCS_CKM.4	-unresolved-
	FMT_MSA.2	-unresolved-
FDP_ACC.1	FDP_ACF.1	FDP_ACF.1
FDP_ACF.1	FDP_ACC.1	FDP_ACC.1
	FMT_MSA.3	FMT_MSA.3
FIA_AFL.1	FIA_UAU.1	FIA_UAU.2
FIA_ATD.1	-	-
FIA_SOS.1	-	-
FIA_UAU.2	FIA_UID.1	FIA_UID.2
FIA_UAU.5	-	-
FIA_UID.2	-	-
FMT_MSA.1	[FDP_ACC.1; FDP_IFC.1]	FDP_ACC.1
	FMT_SMR.1	FMT_SMR.1
	FMT_SMF.1	FMT_SMF.1
FMT_MSA.3	FMT_MSA.1	FMT_MSA.1
	FMT_SMR.1	FMT_SMR.1
FMT_SMR.1	FIA_UID.1	FIA_UID.2
FMT_SMF.1	-	-

SFR	Dependencies	Resolution
FPT_RVM.1	-	-
FTP_TRP.1	-	-
FTA_TSE.1	-	-

Table 12: Dependencies of SFRs of TOE

Any unresolved dependencies are justified in the following table:

SFR	Unresolved dependency	Justification
FAU_STG.3a/b FAU_STG.4	FAU_STG.1	The TOE does not require the protection of the audit trail against modification or deletion, because only reading operations are provided. The deletion of audit records is not considered within the scope of usual TOE management operation. Instead, tools of the TOE environment (operating system) must be used, where the protection of the audit trail is supported by the fact that TOE environment administrators are considered trustworthy (A.NOEVIL), and they competently administer the TOE environment (A.ADMIN).
FCS_CKM.1	FCS_CKM.4	Session keys are only valid for the duration of an established session; no subsequent destruction is necessary.
FCS_CKM.1	FMT_MSA.2	The generation of keys without known weaknesses is inherently provided by following the standard specified in FCS_CKM.1.
FCS_CKM.2	FCS_CKM.4	Session keys are only valid for the duration of an established session; no subsequent destruction is necessary.
FCS_CKM.2	FMT_MSA.2	The generation of keys without known weaknesses is inherently provided by following the standard specified in FCS_CKM.2.
FCS_COP.1a	FCS_CKM.4	Session keys are only valid for the duration of an established session; no subsequent destruction is necessary.
FCS_COP.1a	FMT_MSA.2	No security attributes are generated as part of the encryption or decryption following the standards specified in FCS_COP.1a
FCS_COP.1b	FCS_CKM.4	Session keys for hashing are only valid for the duration of an established session; no subsequent destruction is necessary.
FCS_COP.1b	FMT_MSA.2	No security attributes are generated as part of keyed digest generation or verification. following the standards specified in FCS_COP.1b.
FCS_COP.1c	FPT_ITC.1/2 or FCS_CKM.1	The generation of public server key pairs is out of scope for this evaluation. As defined in

SFR	Unresolved dependency	Justification
		A.AUTHKEY, the keys are properly installed. This is done during the setup of the TOE for the evaluated configuration and therefore no explicit key import security function is needed.
FCS_COP.1c	FCS_CKM.4	The destruction of public server key pairs is out of scope for this evaluation.
FCS_COP.1c	FMT_MSA.2	No security attributes are generated as part of the signature generation.
FCS_COP.1d	FPT_ITC.1/2 or FCS_CKM.1	No keys are generated or imported because digest generation and verification works without using keys.
FCS_COP.1d	FCS_CKM.4	No key material is used for generating or verifying message digests.
FCS_COP.1d	FMT_MSA.2	No security attributes are generated as part of digest generation or verification.

Table 13: Justification of unresolved SFR dependencies

The following table lists the dependencies of SFRs in the TOE environment and their resolution:

SFR	Dependencies	Resolution
FIA_ATD.1	-	-
FIA_UAU.2	FIA_UID.1	FIA_UID.2(e)
FIA_UAU.4	-	-
FIA_UID.2	-	-
FPT_STM.1	-	-
FCS_RNG.1	-	-

Table 14: Dependencies of SFRs for the TOE environment

8.2.5 Internal Consistency and Mutual Support of SFRs

Users want to use services to which access is controlled by the TOE. To request access to any service, the user must identify (FIA_UID.2) and authenticate (FIA_UAU.2 for password authentication) himself to the TOE. A sufficient quality of passwords is enforced by FIA_SOS.1. This requires that the appropriate security attributes are associated to a user (FIA_ATD.1). Measures against brute force attacks that target the authentication mechanisms are required (FIA_AFL.1). Additional denying of session establishments are specified by FTA_TSE.1 based on user attributes specified by FIA_ATD.1. Default values for security attributes related to to the FDP_ACC.1/FDP_ACF.1 are specified by FMT_MSA.3. FIA_UAU.5 requires that the TOE can handle password and SecurID authentication attempts. Authentication using secure one-time credentials, performed by SecurID servers in the TOE environment, are required FIA_UID.2(e), FIA_UAU.2(e) and FIA_UAU.4(e). This requires that the appropriate security attributes are associated to a user as specified by FIA_ATD.1(e).

The TOE controls access to services with the help of access control policy (FDP_ACC.1). These policy defines roles and rules for subjects, objects and allowed operations through access regulations that are specified by FDP.ACF.1. Security attributes used by access control mechanisms must be correctly associated to user subjects as specified by

FIA_ATD.1. Access control is always enforced and applies to all relevant actions in respect to the SFPs as specified by FPT_RVM.1.

Information for authentication and authorization is securely transmitted over a trusted path (FTP_TRP.1) that enables confidentiality and integrity through cryptographic operations. Here, signatures are generated (FCS_COP.1c using FCS_COP.1d for compressing the data input), symmetric keys are securely generated and exchanged (FCS_CKM.2), messages are encrypted with the exchanged symmetric keys (FCS_COP.1a) and messages are integrity protected through keyed hashing also using the exchanged symmetric keys (FCS_COP.1b, FCS_COP.1d). The generation of the symmetric keys is supported by FCS_RNG.1(e), which ensures that random numbers are generated using sufficient entropy. It was chosen to define random number generation explicitly, because Part 2 of the Common Criteria do not contain generic security functional requirements for Random Number generation.

Different security-relevant events can occur. These events can be logged together with relevant information (FAU_GEN.1, FPT_STM.1 (e)), which enables a later audit. This includes tracing back an action to the originator, which is supported by FAU_GEN.2. The collected data can be read by authorized users (FAU_SAR.1). Prevention of audit data loss in case of low audit storage space is specified by FAU_STG.3a/b and FAU_STG.4.

The specification of management functions and the security attributes are required by FMT_SMF.1 and FMT_MSA.1, and controlled access to these functions for defined security roles is required by FMT_SMR.1.

8.2.6 Security Assurance Requirements Dependencies Analysis

The Common Criteria authors have ensured that EAL2 is a sound selection of assurance components where all dependencies have been resolved. Since the augmentation of ALC_FLR.1 does not have any dependencies, there is no need to verify the consistency of the assurance component selection.

8.2.7 Evaluation assurance level and Strength of Function

The evaluation assurance level 2 augmented by ALC_FLR.1 (EAL2+) was chosen to provide low to moderate level of independently-assured security. The assurance classes of EAL2+ address obvious vulnerabilities (e.g., public domain), which is consistent with the assumption that remote users of the TOE originate from a well-managed user community, and that the TOE is not intended to be used in an environment in which the value of information and resources in the protected network is high, which implies that the attack motivation is low. The chosen assurance level ensures that the TOE provides adequate protection against casual breach of TOE security by attackers trying to exploit obvious vulnerabilities.

A Strength of Function claim of SOF-medium was only made for the password authentication mechanism for FIA_SOS.1, which is consistent with the EAL and the intended environment.

No strength of function analysis is performed for the cryptographic algorithms supported by the TOE as well as the process of the generation of the keys used by those cryptographic algorithms.

8.3 TOE Summary Specification Rationale

The TOE IT security functions work together to satisfy the security functional requirements. Below, a justification is presented for each SFR, how the related security functions meet the requirements and for the sum of SARs, as well.

8.3.1 Security Functions Justification

The following table shows that the IT security functions (SF), as specified in the TOE Summary Specification, meet all the security functional requirements (SFR) for the TOE and work together to satisfy the TOE security functional requirements.

SFR	Security Functions (TOE Summary Specification)
FAU_GEN.1	The generation of audit records is satisfied by SF.AU.1, except for the function of generating audit logs in case the audit function stops working. SF.AU.1 still provides a reasonable audit functionality, because in case the audit function is not working, no further actions that require logging are possible. This functionality implicitly logs, by its behaviour, that the audit system is not working, which is also indicated to users (no actions are possible).
FAU_GEN.2	The association of user identity and the audit event is satisfied by SF.IA.10. The audit data including the subject identity is then generated by SF.AU.1.
FAU_SAR.1	Read access to the log records is satisfied by SF.AU.5
FAU_STG.3a/b	Requirements of limits of audit trails and notifications is satisfied by SF.AU.3/4. Configuration of notification is satisfied by SF.MGMT.1.
FAU_STG.4	Actions in case of a full audit trail are defined by SF.AU.2
FCS_CKM.1	Key generation for AES and HMAC-SHA-1 is satisfied by SF.CRYPTO.1 which describes the actions performed for the Diffie-Hellman key exchange. Therefore, SF.CRYPTO.1 satisfies both FCS_CKM.1 and FCS_CKM.2.
FCS_CKM.2	Key distribution is also covered by SF.CRYPTO.1 which describes the actions performed for the Diffie-Hellman key exchange. Therefore, SF.CRYPTO.1 satisfies both FCS_CKM.1 and FCS_CKM.2.
FCS_COP.1a	Encryption and decryption are described in SF.CRYPTO.3
FCS_COP.1b	Functions for keyed hash generation and verification are described in SF.CRYPTO.4
FCS_COP.1c	Signature generation is described in SF.CRYPTO.2
FCS_COP.1d	Digest generation and verification is implied by SF.CRYPTO.4 and mentioned by SF.CRYPTO.2.
FDP_ACC.1/ FDP_ACF.1	The role, folder, service, component descriptions are covered by SF.AC.1/2. The effect of the associations between objects is described by SF.AC.2. The effect of access rules are described in SF.AC.1/2/4. The administrative access is defined by SF.AC.5.
FIA_AFL.1	Handling of failed authentication attempts is described in SF.IA.11

SFR	Security Functions (TOE Summary Specification)
FIA_ATD.1	The existence of user ID and password is satisfied by SF.IA.4, which uses them for authentication. Permitted authentication methods are checked by SF.IA.3. A password change during authentication has to comply to the password policy as satisfied by IA.5. SF.IA.9 describes the use of the attributes to force the user to change his password or wars him that the password is about to expire. Expired user accounts are checked in SF.IA.12. Permitted user roles are checked by AC.1. The verification of allowed authentication methods is satisfied by SF.IA.3. Management of user attributes is specified in SF.MGMT.6-8, where the management of passwords must comply to the password policy. The attribute, that defines whether the user is allowed to change his own password, is mentioned in SF.MGMT.9. The management of all other attributes is covered by SF.MGMT.7.
FIA_SOS.1	The enforcement of the quality metric for passwords is satisfied by SF.IA.5.
FIA_UAU.2	Each user is authenticated as described in SF.IA.4.
FIA_UAU.5	The recognition of different authentication methods is described in SF.IA.3. Password authentication is performed by the TOE as described in SF.IA.4. The delegation of SecurID authentication is described SF.IA.8.
FIA_UID.2	Each user is identified as described in SF.IA.1/2.
FMT_MSA.1	Management of security attributes is covered by: • SF.MGMT.6/7/9 (user attributes) • SF.MGMT.3/4 (password policy attributes) • SF.MGMT.10 (access rules) • SF.MGMT.13/14 (associations between roles/folders/services/components) • SF.MGMT.5 (SecurID server connection configuration) • SF.IA.5 (users may be allowed to change their own passwords during SSH authentication)
FMT_MSA.3	Management of associations and objects is described by SF.MGMT.10-13. The list of roles for a user is specified by SF.MGMT.6.
FMT_SMR.1	Management of security roles is part of SF.MGMT.12, which also covers the administrative component that must be associated to a role (via folder/service structure), in order to grant administrative access to a user of that role.
FMT_SMF.1	TOE management functions are specified in SF.MGMT.1-13.
FPT_RVM.1	Continuous access control is satisfied through SF.IA.1-10 and SF.AC.1-5. This satisfies the requirement that the TSP enforcement functions are invoked and succeed before allowing any other function in the TSC. The TSP enforcement functions consist of the security functions for identification and authentication, and the access control functions.

SFR	Security Functions (TOE Summary Specification)
FTA_TSE.1	The denying of session establishments between a remote user and the TOE is satisfied by SF.IA.2 (identification), SF.IA.6 (authentication) and SF.IA.12 (expired user account).
FTP_TRP.1	The trusted path is satisfied by SF.CRYPTO.1-4.

Table 15: Security Function Rationale

8.3.2 Mutual support of security functions

SF.CRYPTO ensures a secure trusted path, featuring server authenticity, confidentiality and integrity between remote users and the TOE. Using the trusted path, remote users can authenticate to the TOE as ensured by SF.IA and get controlled access to services per SF.AC.

Access of administrative users is also mediated through SF.AC and enables the management of security functions as specified by SF.MGMT.

Security-relevant events are logged as described in SF.AU. Logged events are associated to a user with the help of SF.IA.

8.3.3 Assurance Measures Rationale

In section 6.2 the TOE summary specification includes a justification that the TOE security assurance requirements are met by the assurance measures.

8.3.4 Minimum Strength of Function Rationale

For the security function SF.IA.5, SOF-medium is claimed for the authentication mechanism. This is done in accordance with the strength of function claim for the corresponding security functional requirement FIA_SOS.1.

The SOF-medium claim does not apply to the cryptographic algorithms, the process of generating keys for those cryptographic algorithms (including the random number generator), or the cryptographic hash functions implemented in the TOE. Excluding cryptographic algorithms and related functions from the strength of function analysis is in compliance with the [CEM], remarks on ASE_REQ.1.15, paragraph 424.

8.4 PP Claims Rational

No claims to any protection profiles are made.

9 Appendix

A.1 Abbreviations

AES	Advanced Encryption Standard
CBC	Cipher Block Chaining
CC	Common Criteria
CCMB	Common Criteria Maintenance Board
CVS	Concurrent Versions System
DNS	Domain Name System
DSS	Digital Signature Standard
EAL	Evaluation Assurance Level
EIC	European Information Centre
FTP	File Transfer Protocol
HMAC-SHA1	Keyed-hash message authentication using SHA-1
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
ISO	International Organisation for Standardization
IT	Information Technology
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
NetBIOS	Network Basic Input/Output System
OS	Operating System
PP	Protection Profile
RADIUS	Remote Authentication Dial In User Service
SF	Security Function
SFP	Security Function Policy
SHA-1	Secure Hash Algorithm version 1
SNMP	Simple Network Management Protocol
SOF	Strength of Function
SSHv2	Secure Shell version 2
ST	Security Target
TCP	Transmission Control Protocol
TOE	Target of Evaluation
TSC	TSF Scope of Control
TSF	TOE Security Functions
TSFI	TSF Interface
TSP	TOE Security Policy

UDP	User Datagram Protocol
VPN	Virtual Private Network

A.2 Glossary

Application-level gateway –- Security component that augments a firewall or NAT employed in a computer network.

Access control –- The access from subjects to objects controlled by the TOE.

Assets — Information or resources to be protected by the countermeasures of a TOE.

Assignment — The specification of an identified parameter in a component.

Assurance — Grounds for confidence that an entity meets its security objectives.

Attack potential — The perceived potential for success of an attack, should an attack be launched, expressed in terms of an attacker's expertise, resources and motivation.

Augmentation — The addition of one or more assurance component(s) from Part3 to an EAL or assurance package.

Authentication data — Information used to verify the claimed identity of a user.

Authorized user — A user who may, in accordance with the TSP, perform an operation.

Class — A grouping of families that share a common focus.

Component — The smallest selectable set of elements that may be included in a PP, an ST, or a package.

Connectivity — The property of the TOE which allows interaction with IT entities external to the TOE. This includes exchange of data by wire or by wireless means, over any distance in any environment or configuration.

Dependency — A relationship between requirements such that the requirement that is depended upon must normally be satisfied for the other requirements to be able to meet their objectives.

Element — An indivisible security requirement.

Evaluation — Assessment of a PP, an ST or a TOE, against defined criteria.

Evaluation Assurance Level (EAL) — A package consisting of assurance components from Part 3 that represents a point on the CC predefined assurance scale.

Evaluation authority — A body that implements the CC for a specific community by means of an evaluation scheme and thereby sets the standards and monitors the quality of evaluations conducted by bodies within that community.

Evaluation scheme — The administrative and regulatory framework under which the CC is applied by an evaluation authority within a specific community.

Extension — The addition to an ST or PP of functional requirements not contained in Part2 and/ or assurance requirements not contained in Part 3 of the CC.

External IT entity — Any IT product or system, untrusted or trusted, outside of the TOE that interacts with the TOE.

Family — A grouping of components that share security objectives but may differ in emphasis or rigour.

Formal — Expressed in a restricted syntax language with defined semantics based on well-established mathematical concepts.

Human user — Any person who interacts with the TOE.

Identity — A representation (e.g., a string) uniquely identifying an authorized user, which can either be the full or abbreviated name of that user or a pseudonym.

Informal — Expressed in natural language.

Instant messaging –- A form of real-time communication between two or more people based on typed text.

Internal communication channel — A communication channel between separated parts of TOE.

Internal TOE transfer — Communicating data between separated parts of the TOE.

Inter-TSF transfers — Communicating data between the TOE and the security functions of other trusted IT products.

IP Tunneling Driver –- Consists of a service and a virtual network adapter which will tunnel IP traffic to and from the AppGate server over the SSHv2 connection. The AppGate IP Tunneling Driver is optional, e.g., if servers behind the AppGate server need to initiate connections to the client computer.

Iteration — The use of a component more than once with varying operations.

Java Web Start –- A framework developed by Sun Microsystems which allows application software for the Java Platform to be started directly from the Internet using a web browser.

Object — An entity within the TSC that contains or receives information and upon which subjects perform operations.

OpenSSH –- A set of free software computer programs implementing the SSHv2 protocol and providing encrypted communication sessions over a computer network.

Organizational security policies — One or more security rules, procedures, practices, or guidelines imposed by an organization upon its operations.

Package — A reusable set of either functional or assurance components (e.g., an EAL), combined together to satisfy a set of identified security objectives.

Pipes (Named pipes) –- Named pipes allow the communication between two processes within a system, using files as access points.

Product — A package of IT software, firmware and/or hardware, providing functionality designed for use or incorporation within a multiplicity of systems.

Protection Profile (PP) — An implementation-independent set of security requirements for a category of TOEs that meet specific consumer needs.

Reference monitor — The concept of an abstract machine that enforces TOE access control policies.

Reference validation mechanism — An implementation of the reference monitor concept that possesses the following properties: it is tamperproof, always invoked, and simple enough to be subjected to thorough analysis and testing.

Refinement — The addition of details to a component.

Roaming –- The client can temporarily suspend the communication with the server. This is useful for unreliable connections like GPRS or if the client computer changes IP address (e.g. a laptop moving to another location).

Role — A predefined set of rules establishing the allowed interactions between a user and the TOE.

RSA ACE/Server –- RSA authentication manager used for SecurID authentication (formerly ACE/Server)

Secret — Information that must be known only to authorized users and/or the TSF in order to enforce a specific SFP.

SecurID –- RSA SecurID is a mechanism developed by RSA Security for performing two-factor authentication to a user to a network resource.

Security attribute — Information associated with subjects, users and/or objects that is used for the enforcement of the TSP.

Security Function (SF) — A part or parts of the TOE that have to be relied upon for enforcing a closely related subset of the rules from the TSP.

Security Function Policy (SFP) — The security policy enforced by an SF.

Security objective — A statement of intent to counter identified threats and/or satisfy identified organisation security policies and assumptions.

Security Target (ST) — A set of security requirements and specifications to be used as the basis for evaluation of an identified TOE.

Selection — The specification of one or more items from a list in a component.

Semiformal — Expressed in a restricted syntax language with defined semantics.

Strength of Function (SOF) — A qualification of a TOE security function expressing the minimum efforts assumed necessary to defeat its expected security behaviour by directly attacking its underlying security mechanisms.

SOF-basic — A level of the TOE strength of function where analysis shows that the function provides adequate protection against casual breach of TOE security by attackers possessing a low attack potential.

SOF-medium — A level of the TOE strength of function where analysis shows that the function provides adequate protection against straightforward or intentional breach of TOE security by attackers possessing a moderate attack potential.

SOF-high — A level of the TOE strength of function where analysis shows that the function provides adequate protection against deliberately planned or organized breach of TOE security by attackers possessing a high attack potential.

Subject — An entity within the TSC that causes operations to be performed.

System — A specific IT installation, with a particular purpose and operational environment.

Target of Evaluation (TOE) — An IT product or system and its associated administrator and user guidance documentation that is the subject of an evaluation.

TOE resource — Anything useable or consumable in the TOE.

TOE Security Functions (TSF) — A set consisting of all hardware, software, and firmware of the TOE that must be relied upon for the correct enforcement of the TSP.

TOE Security Functions Interface (TSFI) — A set of interfaces, whether interactive (man-machine interface) or programmatic (application programming interface), through which TOE resources are accessed, mediated by the TSF, or information is obtained from the TSF.

TOE Security Policy (TSP) — A set of rules that regulate how assets are managed, protected and distributed within a TOE.

TOE security policy model — A structured representation of the security policy to be enforced by the TOE.

Transfers outside TSF control — Communicating data to entities not under control of the TSF.

Trusted channel — A means by which a TSF and a remote trusted IT product can communicate with necessary confidence to support the TSP.

Trusted path — A means by which a user and a TSF can communicate with necessary confidence to support the TSP.

TSF data — Data created by and for the TOE, that might affect the operation of the TOE.

TSF Scope of Control (TSC) — The set of interactions that can occur with or within a TOE and are subject to the rules of the TSP.

User — Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.

User data — Data created by and for the user, that does not affect the operation of the TSF.

X.509 certificates –- Certificates conformant to the X.509 public key infrastructure standard (PKI).